

A Novel Predictor Based Framework to Improve Mobility of High-Speed Teleoperated Unmanned Ground Vehicles

by

Yingshi Zheng

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Mechanical Engineering)
in The University of Michigan
2018

Doctoral Committee:

Associate Research Scientist Tulga Ersal, Co-Chair
Professor Jeffrey L. Stein, Co-Chair
Dr. Mark J. Brudnak, TARDEC
Professor Brent Gillespie
Dr. Paramsothy Jayakumar, TARDEC
Professor Ilya V. Kolmanovsky

Yingshi Zheng
zhengys@umich.edu
ORCID iD: 0000-0002-2591-4293

© Yingshi Zheng 2018
All Rights Reserved

This dissertation is dedicated to my family.

ACKNOWLEDGEMENTS

First of all, I would like to express my sincere gratitude to my advisors, Prof. Jeffrey L. Stein and Dr. Tulga Ersal for their consistent mentorship and support through my research journey. Prof. Stein has been very helpful in guiding me to form a thorough and well-founded research. His kindness and willingness to foster my research skills with his rich research experience has encouraged me to devote myself to the academic world. I have learned a lot from his constructive advices on how to tailor the research goals and convey them in a clear and compact way. I cannot thank Dr. Ersal enough for his consistent responsibility and support as a mentor. He is very responsible in following my research and education progress, discussing technical details and difficulties in my research and providing essential feedback to me. He is also very patient in helping me improve my presentation skills and technical writing skills on papers and posters. It has been a truly great honor and unforgettable experience under their mentorship.

Working with Dr. Paramsothy Jayakumar and Dr. Mark J. Brudnak is a pleasure. I deeply appreciate their insightful comments and timely feedback provided on my work. I also want to convey my thanks to the rest of my committee members, Prof. Brent Gillespie and Prof. Ilya V. Kolmanovsky, for bringing their expertise and providing valuable suggestions towards my research.

For the beloved university I spent six years in, the University of Michigan, I would like to thank the Automotive Research Center for providing the funding for this research and Rackham Graduate School, college of engineering and department of

mechanical engineering for the conference travel funds and support on my education.

Being a member of Automated Modeling Lab, I would like to thank all the colleagues, past and present - Dr. Jiechao Liu, Dr. Xinyi Ge, Dr. Xin Zhou, Huckleberry Febbo, Alireza Goshtasbi, Dr. Hossein Mirinejad, John Wurts, James Dallas, Kshitij Jain, Yifan Weng, Yue Tang and Jiahui Fu, for their friendship and valuable suggestions on my research and presentation. I have learned a lot from them and really enjoy the atmosphere of having heated discussion on various research topics of interest.

I really appreciate Dr. Xinyi Ge for his contribution and valuable suggestions on analyzing the predictor steady state performance in frequency domain. I would like to thank Prof. Gillespie again for providing the Mini Baja Vehicle platform and his students, Lars Watts and Dr. Amirhossein Ghasemi for their great help on modifying the platform to make the field test of the predictors implemented on the Mini Baja Vehicle achievable.

Also, I really appreciate the care and great help from my friends. Thank you very much on helping me reach where I am now and I hope you are also pursuing what you intend to.

Finally, I would like to thank my parents, Fanghui Zheng and Guofen Ying, my grandparents and all family members. Their support and love are the driving force that makes me insisting on these years of study, which is what I really want to devote myself to. This work would not have been possible without them.

TABLE OF CONTENTS

DEDICATION	ii
ACKNOWLEDGEMENTS	iii
LIST OF FIGURES	viii
LIST OF TABLES	xii
ABSTRACT	xiii
CHAPTER	
I. Introduction	1
1.1 Motivation	1
1.2 An Overview of Methods to Address Communication Delays .	3
1.3 Research Objective and Questions	7
1.4 Organization of the Dissertation	8
II. Background	9
2.1 Vehicle Teleoperation System	9
2.2 Communication Delays	10
2.3 Predictive Display	12
2.3.1 Prediction Methods Predicting Vehicle States	14
2.3.2 Display Methods	19
2.4 Prediction of the Human Operator Commands	20
2.5 Summary of Existing Prediction Based Methods	22
III. A Novel Model-Free Predictor for Delay Compensation . . .	24
3.1 Predictor Dynamics	25
3.2 Predictor Stability Analysis	28
3.2.1 Constant Delay Case	29

3.2.2	Varying Delay Case	30
3.3	Frequency Domain based Performance Analysis	38
3.4	Case Study	42
3.4.1	Performance versus Input Frequencies	45
3.4.2	Performance with Simulated Delays	49
3.4.3	Performance with Hardware Networks	55
3.5	Closed-Loop System Stability with Predictors	57
3.5.1	“Mixed” Passivity and Small Gain Method	57
3.5.2	Simulation Results	61
3.6	Saturation and Resetting Scheme	63
3.7	Predictor Design Procedure	67
IV.	A Predictor Based Framework with Blended Prediction Architecture	71
4.1	Structure of a Novel Predictor Based Framework	71
4.2	Predictor Design	76
4.3	Feedforward Branch	80
4.4	Open-Loop Evaluation of Heading Prediction Accuracy	83
V.	Simulation Based Human-In-the-Loop Test	88
5.1	Simulation Platform	88
5.1.1	Vehicle Platform	88
5.1.2	Driver Station	90
5.1.3	Platform Capability	91
5.2	Experiment Design	92
5.2.1	Simulated delays	92
5.2.2	Parameters in the Predictor Based Framework	97
5.2.3	Test Details	98
5.2.4	Test Procedure	100
5.3	Analysis Methods	102
5.4	Results	103
5.5	Discussion	107
VI.	Field Demonstration and Testing	109
6.1	Vehicle Platform and Driver Station	109
6.2	Steering Command Prediction	111
6.3	Vehicle State and Camera View Prediction	114
6.3.1	Vehicle State Recognition	118
6.3.2	Vehicle State Prediction and Predicted Camera View	119
6.4	Preliminary Results of Closed-Loop Experiments	120
6.5	Current Issues and Future Work	123
6.5.1	Video Transmission Quality	123

6.5.2	Vehicle Speed Control	125
6.5.3	Steering Haptic Feedback	125
VII.	Conclusions and Future Work	127
7.1	Dissertation Summary	127
7.2	Conclusions and Original Contributions	128
7.3	List of Publications	130
7.4	Future Work	131
7.4.1	Compensating Visual Delays for Semi-Autonomous UGVs	131
7.4.2	Theoretical Development of Predictors	132
7.4.3	Field Test With Mini Baja Vehicle	133
BIBLIOGRAPHY	134

LIST OF FIGURES

Figure

1.1	A general paradigm of vehicle teleoperation.	2
2.1	In the teleoperated vehicle system, control commands and vehicle information in terms of states and camera views are communicated with control delays $\tau_1(t)$ and sensor delays $\tau_2(t)$, respectively. A round trip delay of $\tau_{RTT}(t) = \tau_1(t) + \tau_2(t)$ deteriorates the vehicle mobility and could even destabilize the system.	10
2.2	The setup of predictive display methods. Vehicle states are predicted considering the round trip delays and a display based on predicted states is therefore generated, showing the instantaneous vehicle outcome under the human operator's control at the Driver Station. . .	13
3.1	Setups of a generic one-way communication from System 1 to System 2 (a) without predictor and (b) with predictor. The novel predictor is placed at the receiver side to compensate delays and generate prediction of the original signal $y(t)$	25
3.2	Stability region of complex λ to ensure asymptotic stability of (3.9).	29
3.3	Above is the round-trip delay measurement between Ann Arbor, USA and Shanghai, China (AA-SH). Below is the generated delay sequence using the two-time-scale Markov Chain. Mean and standard deviation of the delays between the measured data and generated delay sequence are close to each other.	33
3.4	Above and below are the histograms of the measured and generated delay sequences. The distributions of delays are close to each other.	35
3.5	The histogram of round-trip delay measurements from three different networks.	36
3.6	Range of predictor parameter λ to ensure a stable predictor. The theorem based on the two-time-scale Markov chain provides a sufficient, yet not very conservative, range of the parameter λ	37
3.7	Gain of (3.23), $M(\omega)$, with fixed $\tau = 0.03$ s for various λ values within the stability range. Predictor bandwidth ω_p increases with larger λ .	40
3.8	Gain of (3.23), $M(\omega)$, with fixed $\lambda = 7.9 = 0.15\lambda_{\max}(0.03)$ for various constant delay values. Predictor bandwidth ω_p increases with smaller τ	41

3.9	Networked motor-shaft-gear system with bilateral communication delays $\tau(t)$. Coupling signals are the shaft torque T_s and the shaft speed ω_s	42
3.10	Bode plot of Subsystem 1 (from E to y_1 and from ω_s to y_1) and Subsystem 2 (from T_s to y_2). Subsystem 1 has cutoff frequency of 2.5 rad/s for input E	44
3.11	Subsystem 2 predictor performance for $\omega = 1.5$ rad/s	48
3.12	Subsystem 2 predictor performance for $\omega = 50$ rad/s	48
3.13	Output trajectory for $\omega = 50$ rad/s. Predictors are capable of improving closed-loop performance under high frequency input.	49
3.14	Comparison of the output trajectory between ideal case and delayed cases with different delays. Larger delay has worse impact on the system.	50
3.15	Normalized performance metrics p_n with different constant delays. Smaller p_n indicates better performance and higher fidelity in networked integration. For all delays, better performance is achieved with the predictors compared to the delayed case.	51
3.16	Output trajectory comparison among ideal case, delayed case with delays of 135 ms, and predictor case. Predictor cases with all λ lead to better performance compared to delayed case.	52
3.17	Normalized performance metrics p_n with different varying delays. Smaller p_n indicates better performance and higher fidelity in networked integration. For all delays, predictors with different λ improve the performance compared to the delayed case.	53
3.18	There exists fast varying jitters in the coupling error of shaft speed signal $y_2(t)$. Jitters are attenuated by Subsystem 2 predictor and the prediction error is smooth, resulting in a smooth predictor output without jitters.	54
3.19	Comparison of shaft speed signals among undelayed $y_2(t)$, delayed $y_2(t - \tau(t))$, and Subsystem 2 predictor output $\hat{y}(t)$. Predictor has the potential of alleviating the jitters in the delayed signal and generating a smooth prediction $\hat{y}(t)$	55
3.20	The histogram of one-way delay measurements (a) from Server to Client (b) from Client to Server. Round trip delay is 90.3 ± 9.3 ms.	56
3.21	p_n with a real MI-CA network. Smaller p_n indicates better performance and higher fidelity in networked integration. Predictors with various λ improve the performance compared to the delayed case.	57
3.22	Interconnection of two nonlinear subsystems M_1 , M_2	58
3.23	Bode plots of M_1 and M_2 (a) without predictors (b) with subsystem 2 predictor $\lambda = 0.60\lambda_{\max}(0.2)$. Predictor extends the frequency bandwidth within which the system is passive.	60
3.24	The product of gain $ M_1(j\omega) \cdot M_2(j\omega) $ (a) without predictors (b) with subsystem 2 predictor $\lambda = 0.60\lambda_{\max}(0.2)$. Predictor slightly amplify the system gain.	61

3.25	When the delays make the closed-loop system unstable, predictors manage to stabilize the system and improve the fidelity as well. . .	63
3.26	Comparison of different heading signals. When applying a saturation and resetting scheme with the predictor, the overshoot in the transient is reduced and thus leads to better predictor transient performance.	65
3.27	There exists oscillation in the predictor output when compensating large delays of 0.6 s with large λ . Applying the saturation and resetting scheme helps reduce the oscillation introduced by the predictor.	66
4.1	A novel prediction based framework to compensate delays.	72
4.2	The blended architecture is composed of a <i>feedback branch</i> , a <i>feed-forward branch</i> and a linear blending for vehicle heading predicting. Minimal vehicle information is relied on to generated prediction of vehicle states with robustness.	74
4.3	Coupling errors of vehicle heading based on one set of human driving data in teleoperated vehicles.	76
4.4	The power spectral density (PSD) of $c(t)$ with zoomed-in view. The 99% occupied band is between 0.005 Hz and 0.296 Hz.	77
4.5	$M(\omega)$ with delays (a) $\tau_p = 0.3$ s and (b) $\tau = 0.6$ s. Frequencies within the bandwidth of $c(t)$, i.e. ω_c , are marked. Predictor improves steady state performance over frequencies with $M(\omega)$ less than 0 dB. . . .	78
4.6	Gains and phases of the frequency response of (4.4) (a) with same u and different A (b) with same A and different u . The fitted steering model matches with collected data points well for each pair of $\{A, u\}$.	82
4.7	An open-loop test setup to evaluate the heading prediction accuracy of the blended architecture.	84
4.8	Prediction accuracy metric $(\Delta\psi)_b$ with various α . Compared to without prediction $((\Delta\psi)_0 = 8.88$ rad), blended headings with $\alpha = 0.6$ using blending configuration I and with $\alpha = 0.3$ using blending configuration II have the best prediction accuracy of $(\Delta\psi)_b = 1.31$ rad and $(\Delta\psi)_b = 1.30$ rad, respectively.	86
4.9	The open-loop output of the blended heading $\psi_b(t)$ with $\alpha = 0.6$ is very close to undelayed heading $\psi_u(t)$, indicating better prediction accuracy compared to that of either feedforward heading $\psi_{FF}(t)$ or feedback heading $\psi_{FB}(t)$ alone.	87
5.1	Human-in-the-loop simulation platform with predictor based framework. The vehicle model is a 14 DoF model of a typical military truck. By means of a steering wheel and set of pedals, a human can control the throttle, brake and steering to drive the vehicle. Control and sensor delays can be specified independently. The prediction framework and blended prediction architecture can be activated or deactivated.	89
5.2	The driving interface displays the camera view and current vehicle speed. A hook on the vehicle hood is considered as a reference of the vehicle's center.	91

5.3	The histogram of one way delay measurement data of MI-CA network. It can be fitted using a Generalized Extreme Value distribution with $k = 0.707, \mu = 0.0546, \sigma = 0.0012$	93
5.4	Simulated delay sequences of (a) control delays and (b) sensor delays, generated by the corresponding fitted GEV models. Mean values are marked as red lines.	93
5.5	Designated track, vehicle and landmarks in the virtual environment.	99
5.6	Comparison of three performance metrics with different delay types and prediction methods.	104
6.1	Vehicle teleoperation system is composed of a driver station and a Mini Baja vehicle platform with teleoperated steering control capability.	110
6.2	One-way prediction of steering commands from Driver Station to Vehicle Platform.	111
6.3	One-way prediction of vehicle states and camera view from Vehicle Platform to Driver Station.	114
6.4	The vehicle response interface is composed of a text region with vehicle states and system time, and a camera view region.	117
6.5	Difference between clear texts and unclear texts. Unclear text regions are marked with a rectangular box.	118
6.6	Comparison between delayed and predicted camera view.	120
6.7	Comparison among vehicle trajectories of ideal, delayed and predicted cases in a double lane change scenario.	121
6.8	Comparison among vehicle steering wheel angle δ of ideal, delayed and predicted cases in a double lane change scenario.	122

LIST OF TABLES

Table

3.1	Performance metrics for different λ values and excitation frequencies over a simulation time window of 30 s. A smaller metric value indicates better performance.	47
3.2	Performance metrics with different constant delays	50
3.3	Means and standard deviations of the round-trip delays for the three networks data	52
3.4	Performance metrics with different varying delays	53
4.1	Prediction accuracy metric $(\Delta\psi)_{\text{FB}}$ with various predictor settings compared to the baseline accuracy $(\Delta\psi)_0$	80
4.2	Two configurations that blend <i>feedforward branch</i> and <i>feedback branch</i> to predict vehicle heading	85
5.1	Simulated control and sensor delays generated by the fitted GEV models. The means are 0.288 s and 0.632 s, respectively.	96
5.2	Different predictor settings to predict individual signal of interest	97
5.3	Mean values of metrics under different scenarios with delays.	105
5.4	The level of improvement in performance metrics with Pred or Pred-Blend , compared to NoPred . "ns" means that no statistically significant improvement is observed in this experiment.	106

ABSTRACT

Teleoperated Unmanned Ground Vehicles (UGVs) have been widely used in applications when driver safety, mission efficiency or mission cost is a major concern. One major challenge with teleoperating a UGV is that communication delays can significantly affect the mobility performance of the vehicle and make teleoperated driving tasks very challenging especially at high speeds.

In this dissertation, a predictor based framework with predictors in a new form and a blended architecture are developed to compensate effects of delays through signal prediction, thereby improving vehicle mobility performance. The novelty of the framework is that minimal information about the governing equations of the system is required to compensate delays and, thus, the prediction is robust to modeling errors.

This dissertation first investigates a model-free solution and develops a predictor that does not require information about the vehicle dynamics or human operators' motion for prediction. Compared to the existing model-free methods, neither assumptions about the particular way the vehicle moves, nor knowledge about the noise characteristics that drive the existing predictive filters are needed. Its stability and performance are studied and a predictor design procedure is presented.

Secondly, a blended architecture is developed to blend the outputs of the model-free predictor with those of a steering feedforward loop that relies on minimal information about vehicle lateral response. Better prediction accuracy is observed based on open-loop virtual testing with the blended architecture compared to using either

the model-free predictors or the model-based feedforward loop alone.

The mobility performance of teleoperated vehicles with delays and the predictor based framework are evaluated in this dissertation with human-in-the-loop experiments using both simulated and physical vehicles in teleoperation mode. Predictor based framework is shown to provide a statistically significant improvement in vehicle mobility and drivability in the experiments performed.

CHAPTER I

Introduction

1.1 Motivation

Unmanned ground vehicles (UGVs) are vehicles operated without on-board drivers and have been widely used in both military and commercial applications when driver safety, mission efficiency or mission cost is a major concern. Potential applications include reconnaissance, surveillance, route clearing, planetary and mine exploration, and farming and rescue tasks [1, 2]. UGVs span a wide spectrum in their mode of operation from teleoperated to semi-autonomous and fully autonomous. Teleoperation describes the mode in which the UGV has no intelligence to sense and react to its environment and a remote human operator controls all actions of the UGV. In the semi-autonomous mode, the control authority is shared between the human operator and an autonomy module. In the fully autonomous mode the autonomy module is responsible for controlling all actions of the UGV. Despite the rapidly developing technologies of sensors and algorithms for autonomy, UGVs are still far from being operated in fully autonomous mode under all circumstances and human operators are involved in the loop to remotely monitor and control the vehicle operation in situations when UGVs are not capable of completing the mission independently and reliably. At the moment, majority of the UGVs in the Army are teleoperated, and enabling high-speed teleoperation of UGVs is critical to the Army [3, 4]. Therefore,

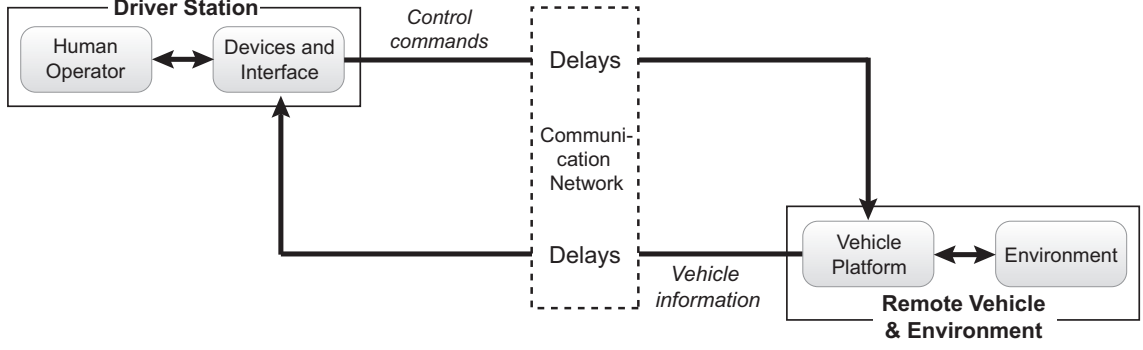


Figure 1.1: A general paradigm of vehicle teleoperation.

this dissertation focuses on high-speed teleoperated UGVs.

A general paradigm of vehicle teleoperation is shown in Fig. 1.1. The driver station is physically separated from the remote vehicle and environment. A human operator at the driver station uses input devices to send through a communication network control commands to the remote vehicle platform that maneuvers in the environment, and receives vehicle information via a human-vehicle interface. Vehicle information usually includes camera views to capture the environment and vehicle states related to its dynamics. As humans get more than 90% of their perception information via vision [2], information presented to the operators through the interface is mostly in the form of visual display.

In teleoperation, reduced situational awareness and the communication delays in the network are the main challenges that reduce vehicle mobility. The former is caused by compromised human perception due to the fact that the operator is not physically in the vehicle, but receives partial information from a remote physical environment [5]. Improving situational awareness is mainly based on design of interfaces to aid human perception and is not the scope of this dissertation. Instead, this dissertation mainly addresses the second challenge, i.e. the communication delays that cause degradation of vehicle mobility.

More specifically, the round trip delays in the network cause asynchrony between

human sending commands to and receiving information from the remote vehicle. When delays are as small as around 130 ms, human operators can adapt to delays with practice, predicting the outcome of their control actions fairly accurately and controlling the vehicle continuously as if there are no delays [6]. However, when delays are larger than this threshold, they can significantly degrade vehicle mobility and even destabilize the closed-loop system, mainly due to operator-induced oscillations when human operators overcompensate for non-negligible errors between their predicted control outcome and delayed response [7, 8, 9]. Delays make vehicle teleoperation very challenging especially at high vehicle speeds, as there is less time for human operators to respond to sudden changes in the environment. For example, in a simulated driving task, human operators were asked to control the vehicle in a lane while maintaining a speed of 55 mph and driving performance was found to be significantly degraded with delays of 170 ms [10].

Therefore, teleoperating vehicles at high speeds under relatively large communication delays is an important research challenge.

1.2 An Overview of Methods to Address Communication Delays

There exist various techniques in the teleoperated vehicle literature to address the challenge of communication delays. For example, a human operator can switch to a move-and-wait control strategy to avoid instability in most of the teleoperation tasks, but this strategy slows down the operation and is error prone [9].

Supervisory control methods rely on various levels of autonomy, so that the operator is only responsible for designating short term objectives and the vehicle autonomously navigates to the objective [11, 12, 13, 14]. With this scheme, delays have no impact on the vehicle control that is handled at the vehicle locally by the automa-

tion. Instead, delays negatively affect the performance by reducing the fidelity for the human operator in monitoring the vehicle and designating the objectives. An event based communication framework helps to deal with this problem [15, 16, 17]. In this framework, packet transmission through the communication channel is handled based on designated events rather than periodically in time. Unless an event is triggered that initiates a packet transmission (for example, updates in commands or vehicle response are necessary for the sake of stability and good estimation performance), driver station and vehicle platform are operated locally relying on simulation of estimated vehicle response and levels of autonomy, respectively. Nevertheless, a key disadvantage remains; namely, supervisory control methods require additional effort and cost to add the necessary level of autonomy to the vehicle.

The above-mentioned methods aim to reduce the frequency of the human operation and the continuous closed-loop teleoperation is replaced by either discrete move-and-wait control or high-level decision making that are affected less by delays. However, doing so either degrades performance or requires additional efforts and costly modifications on the system.

In contrast, prediction based methods aim to compensate delays to improve the performance under continuous teleoperation, and they do so through predicting the vehicle or human operator motions. One well-known approach in this category is the predictive display. In the predictive display scheme, instantaneous vehicle response that is likely to result from the current actions of the operator is predicted and visualized at the driver station, either overlaying onto or replacing the delayed visuals, to reduce the asynchrony between human’s control actions and the subsequent vehicle response. As human operators mostly rely on vision in driving [2], predictive display methods have been evaluated to be very helpful in improving vehicle mobility under communication delays [14, 18, 19, 20, 21, 22].

As another example, the Smith predictor relies on a linear vehicle model to coun-

teract the effects of delay and simplify the closed-loop system transfer function, so that the denominator of the transfer function does not include the delay term, thereby enabling the achievement of closed-loop stability by designing the controller for the system without delay [23, 24].

In [25, 26], human operator’s control commands are predicted into the future before being sent to the remote vehicle to compensate the delays. This prediction is based on a model or assumption of human motion.

Most of these prediction methods are model-based, requiring, for example, an accurate full vehicle model or human model to perform the prediction [14, 18, 19, 20, 23, 24, 25]. However, obtaining an accurate model can be difficult, as modeling all the salient dynamics properly and parameterizing the model correctly can be a challenge. Thus, these model-based predictions may suffer from low robustness to modeling errors.

Alternatively, model-free methods can be employed. Being model free, these methods do not require knowledge of vehicle dynamics or human behavior and are thus robust to modeling errors, but usually have worse prediction performance than model-based methods due to lack of incorporation of any domain-specific knowledge. Examples include predictive display with clothoid prediction of vehicle trajectory [20] and Taylor-series-expansion based Kalman filter prediction [27, 28]. However, they either require additional assumptions on vehicle or operator motions, such as constant speed, or an accurate knowledge or estimation of the statistical characteristics of the noise that drives the filter to estimate the high order derivatives of the delayed signals. The performance of these model-free methods depends on how realistic these assumptions are or how accurately the noise statistics can be known.

Other delay compensation approaches from the telerobotics literature can potentially also be applied to teleoperated vehicles for driving tasks, but their performance in terms of improving vehicle mobility remains to be studied. For example, passivity

based methods, including the wave variable and scattering transform [29, 30, 31], PD-like controller [32, 33], and passive controller with energy dissipation [34, 35, 36], are widely applied in teleoperated manipulation and telesurgery applications, where haptic feedback is more critical than vision for the operators to complete the missions. These methods either render the communication channel passive or dissipate energy to achieve a passive thus stable closed-loop system while improving the performance under delays. In this context, performance is typically characterized by transparency. Transparency is defined as the degree of how well human operators feel the environment when teleoperating compared to when they directly interact with it [37] and is mostly in the sense of haptics in telerobotics. Although some of these methods have been applied on small teleoperated mobile robots with virtual force feedback based on distances to obstacles [38, 39], robot velocities [40, 41], or tire contact force [42], driving teleoperated vehicles especially at high speeds relies more on vision than haptic feedback. Therefore, it is unknown how improving haptic transparency helps with improving vehicle mobility in high-speed vehicle teleoperation with delays [5].

A teleoperated vehicle can also be considered as a distributed hardware-in-the-loop experiment with two nodes. Networked hardware-in-the-loop literature presents methods to ensure a high-fidelity integration with no or minimal knowledge about the system [43, 44, 45, 46, 47], with a focus on the control of communication similar to the telerobotics literature. However, some of these methods are more suitable in an experimental setting, where experiments can be repeated under controlled conditions.

Thus, there exists a gap within prediction-based methods. Model-based predictions are accurate when accurate model is available, but may suffer from low robustness to modeling errors, while existing model-free prediction methods have the benefit of robustness to modeling error, but rely on assumptions that may not always hold or require knowledge about noise statistics that may not be available with needed accuracy. A method does not yet exist that takes both performance and robustness into

consideration when performing prediction on transmitted signals and compensating delays to improve mobility for high speed teleoperated UGVs.

1.3 Research Objective and Questions

Given the gap in the literature identified in the previous section, the objective of this dissertation is to develop a new prediction based framework to compensate communication delays and robustly improve the performance of vehicle mobility for high speed teleoperated UGVs. The key novel feature that this framework is desired to possess is to require minimal information about the governing equations of vehicle dynamics or human motion to benefit from performance robustness to modeling errors. The motivation for this requirement is to enable a solution that is as platform and operator independent as possible and can thus be readily applied to various vehicles with minimal tuning.

To meet this objective, two major research questions are studied in this dissertation:

- How much can the mobility metric of completion time and track keeping error under a track following scenario of a high-speed teleoperated UGV be improved robustly in a vehicle- and operator-agnostic manner with a model-free predictor based framework to compensate the delays?
- How much can the mobility metric of completion time and track keeping error under a track following scenario of a high-speed teleoperated UGV be further improved using the model-free predictor based framework in combination with minimal system information to compensate the delays?

1.4 Organization of the Dissertation

The rest of the dissertation is organized as follows. Chapter II reviews the background on vehicle teleoperation in depth and summarizes the prediction-based delay compensation methods in the literature along with their limitations. Chapter III presents a model-free predictor of new form, along with detailed analysis of its stability and performance. A general predictor design procedure is also presented. Chapter IV presents a predictor based framework including a blended prediction architecture that relies on minimal information of the vehicle lateral response. The prediction accuracy is evaluated in open loop. Chapter V and VI evaluate the mobility performance of teleoperated UGVs with delays and the developed predictor based framework with simulated and physical vehicles, respectively, based on human-in-the-loop experiments. Finally, Chapter VII concludes the dissertation with a summary of contributions in this dissertation and suggests several directions for potential future research.

CHAPTER II

Background

In this chapter, background on generic vehicle teleoperation is provided first. A review of existing prediction based delay compensation methods are addressed, providing motivation for a new model-free predictor in Chapter III and a blending architecture in Chapter IV.

2.1 Vehicle Teleoperation System

The generic paradigm of a teleoperated vehicle is illustrated in Fig. 2.1. The Driver Station where the human operator is located and the Remote Vehicle are two geographically separate subsystems that are coupled through a communication network. While communication delay can be broken down into four components (i.e., processing delay, queuing delay, transmission delay, and propagation delay) [48], here we consider it as a single, pure delay. Delays are considered bilaterally: control delays $\tau_1(t)$ from the Driver Station to the Remote Vehicle and sensor delays $\tau_2(t)$ from the Remote Vehicle to the Driver Station. Thus, when the Driver Station sends out a vector of control commands $\mathbf{y}_1(t)$ (including steering, throttle and brake), it takes a round trip delay of $\tau_{RTT}(t) = \tau_1(t) + \tau_2(t)$ to receive a vector of vehicle states $\mathbf{y}_2(t)$ (usually including heading, location, speed) and camera views $\mathbf{Img}(t)$ as a response from the Remote Vehicle.

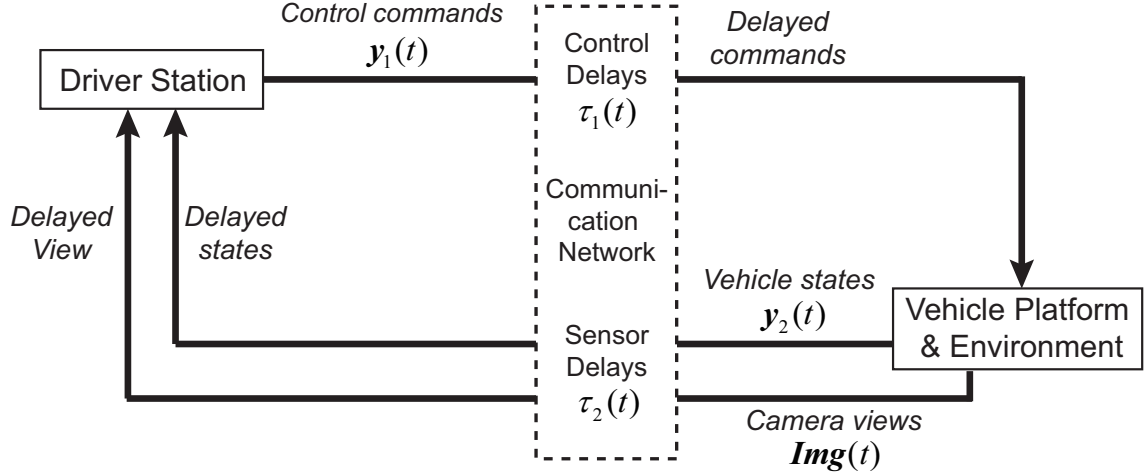


Figure 2.1: In the teleoperated vehicle system, control commands and vehicle information in terms of states and camera views are communicated with control delays $\tau_1(t)$ and sensor delays $\tau_2(t)$, respectively. A round trip delay of $\tau_{RTT}(t) = \tau_1(t) + \tau_2(t)$ deteriorates the vehicle mobility and could even destabilize the system.

2.2 Communication Delays

The amount of communication delay depends on the communication distance and type of network. When the distance is on the order of meters to hundreds of meters, subsystems communicate via a local wireless network such as high frequency radios and WLAN. Vehicles are usually operated at low speeds due to limited range of communication and delays as small as tens of milliseconds are observed. When teleoperation over longer distances are needed, as is the case in military operations, long-distance communication networks are needed. Long-distance communication is more of a concern, because large delays and jitters cause significant degradation in vehicle mobility. Potential networks used for long-range vehicle teleoperation include cellular networks, Internet or satellite networks. The range of delays for these networks are summarized as follows. 3G and 4G LTE mobile network has one-way delay ranging from around 50 ms to spikes of over 1 s [49, 50, 51]. Ultra low latency and high bandwidth 5G mobile network is under development, and at the Mobile World Congress 2017 less than 10 ms of one-way delay was reported from an initial test on

vehicle teleoperation [52]. Delays in wired Internet varies with the distance and the mean delays could reach around 300 ms between different continents [53]. Wireless connection via Internet is more subject to jitter compared to wired connection. In the battlefield where commercial networks are not available, communication is executed via satellite links and delays are on the order of hundreds of milliseconds [54, 55] for cross-country teleoperation. Transmission of camera views requires higher bandwidth than transmission of control and vehicle state signals and delays are expected to be even larger (beyond 1 s).

Note that for a linear system, one-way delays $\tau_1(t)$ and $\tau_2(t)$ have the same effect on the open-loop system output as a single round trip delay of $\tau_{RTT}(t) = \tau_1(t) + \tau_2(t)$, because time shifting with delays is a linear transformation. Inspired by this fact, some researchers simulated $\tau_{RTT}(t)$ unilaterally in simulation-based experiments [18, 56]. The directionality of delays (either from the Driver Station to the Remote Vehicle or from the Remote Vehicle to the Driver Station) was investigated in [57]. Operators felt the vehicle more difficult to control when the delays were simulated from the Remote Vehicle to the Driver Station (i.e. delays in vision), but no significant difference in vehicle mobility was observed. Thus, in this dissertation, degradation of teleoperation performance is mainly discussed based on the magnitude of round trip delays, while establishment of closed-loop nonlinear system stability is still based on bilateral one-way delays.

Most research efforts consider delays as predefined constant values. However, delays are typically varying in the actual world. The relationship of teleoperation performance between constant delays and varying delays has been studied in various works [18, 57, 58, 59]. A track following scenario was tested in [18]. Test results indicated worse performance in lane offset with delays varying between 400 ms and 1100 ms than with constant delays of 700 ms. No significant difference between varying and constant delays was observed for performance of vehicle speed, workload and motion

sickness. In [58, 59], path following performance among varying delays distribution with different means and standard deviations were tested and were equated to the performance with constant delays of larger means. However, more navigation error and less completion time were observed with constant delays than with varying delays in [57] and researchers suggested that this discrepancy might be due to overconfidence in the ability to adapt to constant delays after training, while this confidence was not gained for the case with varying delays. Given these previous findings, this dissertation studies the delay compensation methods against both constant and varying delays.

In this dissertation, round trip delays between 0.3 s and 1.0 s are considered based on the delay range of the common networks. This range is also in agreement with the delay values tested in most human-in-the-loop experiments for teleoperating high speed UGVs in the literature [6, 10, 18, 21, 22, 60, 61]. Sensor delays $\tau_2(t)$ from the Remote Vehicle to the Driver Station are assumed to be larger than control delays $\tau_1(t)$ considering that packets of larger size (including the camera views) are transmitted.

Next, a review of the existing prediction based methods including predictive display and human operation prediction is presented and their advantages and disadvantages are summarized.

2.3 Predictive Display

Predictive display methods are widely used in vehicle teleoperation to reduce the effect of delays on driving performance. It is first studied by Arnold and Braisted for planetary rovers [62]. A general scheme is shown in Fig. 2.2, including two parts, namely, state prediction and image processing. Compared to the general vehicle teleoperation system in Fig. 2.1, where it takes $\tau_{RTT}(t)$ for the human operator to receive the vehicle response corresponding to his/her control commands, vehicle states

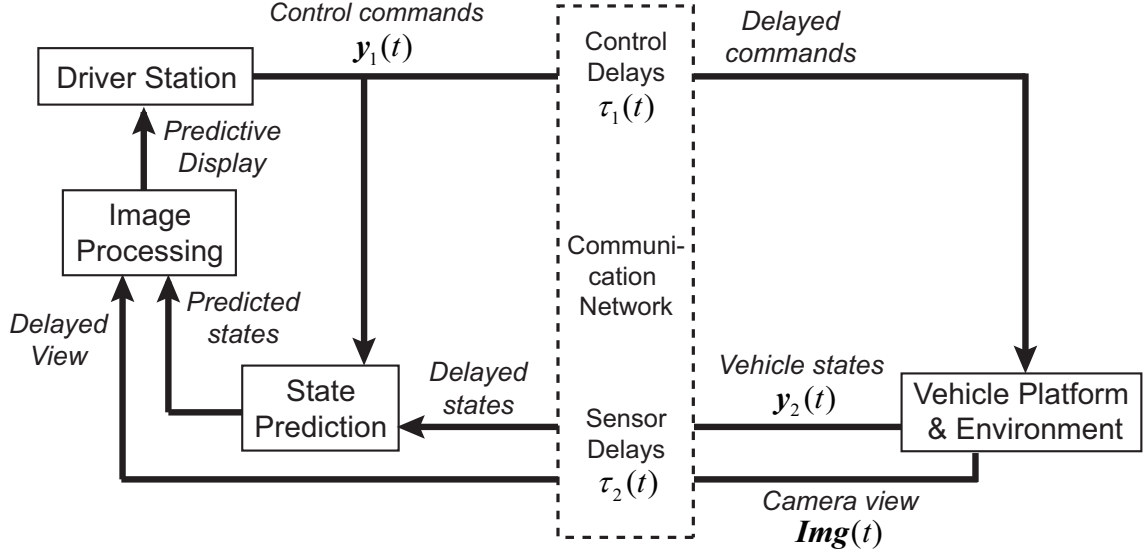


Figure 2.2: The setup of predictive display methods. Vehicle states are predicted considering the round trip delays and a display based on predicted states is therefore generated, showing the instantaneous vehicle outcome under the human operator’s control at the Driver Station.

are predicted to obtain a response as close to the actual response as possible when considering the round trip delay $\tau_{RTT}(t)$. Then, delayed camera views are processed to incorporate the predicted states and a modified display is presented to the human operator. Since the operator can see the direct vehicle outcome to his/her current control commands instantaneously, predictive displays are capable of mitigating the effect of delays on closed-loop teleoperation. For example, in [18], applying predictive display under mean round trip delays of 700 ms improves vehicle speed by 12% and lane keeping accuracy by 26% compared to no prediction. In [22], applying predictive display under constant round trip delays of 500 ms improves vehicle speed by 29% and reduces path deviation error and heading error by 35% and 42%, respectively, compared to without predictive display.

Various prediction methods and display methods can be used in the setup of predictive displays and are summarized as follows.

2.3.1 Prediction Methods Predicting Vehicle States

This section discusses methods available in the literature that can be implemented in the block of “State Prediction” in Fig. 2.2 to predict vehicle states. Depending on whether information about the governing equations of vehicle dynamics or vehicle parameters is required to perform prediction, prediction methods are classified as model-free or model-based.

2.3.1.1 Model-Free Prediction

Model-free prediction of vehicle states is mainly based on geometric constraints on vehicle trajectory and thus is independent of vehicle dynamics or parameters. However, assumptions on vehicle trajectory are usually posed. In the clothoid prediction, the vehicle is assumed to be moving in clothoids while keeping its current longitudinal speed and curvature changing rate constant for the next τ_{RTT} [20]. Clothoids are curves with constant curvature change, which are widely used in road design to ensure a smooth transition between arcs with different radii. It is often used in driver assistance systems to predict smooth vehicle motions with no lateral jerk [63, 64]. The curvature $C_{0,c}$ and curvature changing rate $C_{1,c}$ of vehicle trajectory is calculated based on yaw rate $\dot{\psi}_c$ and longitudinal speed u_c from the vector of received delayed states at t_c :

$$\begin{aligned} C_{0,c} &= \frac{\dot{\psi}_c}{u_c} \\ C_{1,c} &= \frac{C_{0,c} - C_{0,c-1}}{u_c(t_c - t_{c-1})} \end{aligned} \tag{2.1}$$

The vehicle heading ψ_p and position x_p, y_p for a prediction horizon of τ_{RTT} is determined as below:

$$\begin{aligned}\psi_p &= C_{0,c}(u_c \tau_{RTT}) + \frac{1}{2} C_{1,c}(u_c \tau_{RTT})^2 \\ x_p &= \sum_{\tau=0}^{\tau_{RTT}} u_c \cos(C_{0,c}(u_c \tau) + \frac{1}{2} C_{1,c}(u_c \tau)^2) \Delta t \\ y_p &= \sum_{\tau=0}^{\tau_{RTT}} u_c \sin(C_{0,c}(u_c \tau) + \frac{1}{2} C_{1,c}(u_c \tau)^2) \Delta t\end{aligned}\tag{2.2}$$

Since there is no closed form for continuously integrating the clothoid curve, x_p, y_p are calculated based on numerical integration with fixed time step Δt . Note that ψ_p, x_p, y_p are with respect to the coordinates of delayed states and represent the difference between predicted and delayed states, as required by some display methods to perform image processing. Clothoid prediction relies on the geometric continuity of vehicle trajectory and is independent of human commands. It is reported in [20] that its open loop prediction performance degrades with large steering wheel angle changes in the prediction horizon τ_{RTT} .

Other model-free prediction methods use high order derivatives of the signals to predict the signals into the future. They are more general methods and could be used to predict any signal including vehicle states. In these methods, Taylor series expansion of a signal $s(t)$ at time t_c up to order N is considered:

$$s(t_c + \Delta t) = s(t_c) + \sum_{i=1}^N s^{(i)}(t_c) \frac{(\Delta t)^i}{i!} + \text{H.O.T.}\tag{2.3}$$

Because there exists noise in the measurement of high order derivatives or derivatives are not available from measurements, Kalman filter based algorithms are used to estimate them [65, 28]. For example, in target tracking applications [27], remote targets are captured in vision with round trip delay τ_{RTT} and there is no direct information of target's speed and acceleration. Target motion needs to be predicted

ahead to compensate the effect of delays. Rewriting Eq. (2.3) in the form of a discrete state space model yields:

$$x[i+1] = \begin{bmatrix} 1 & \Delta t & \cdots & \frac{(\Delta t)^N}{N!} \\ & 1 & \cdots & \frac{(\Delta t)^{N-1}}{(N-1)!} \\ & & \ddots & \vdots \\ & & & 1 \end{bmatrix} x[i] + \begin{bmatrix} \frac{(\Delta t)^{N+1}}{(N+1)!} \\ \frac{(\Delta t)^N}{N!} \\ \vdots \\ \Delta t \end{bmatrix} u[i] \quad (2.4)$$

where $x[i]$ is the state to estimate and includes the derivatives of $s(t)$ at time step i , i.e., $x = [s \ s^{(1)} \ \cdots \ s^{(N)}]$ and $u[i]$ is the high order term in Eq. (2.3) treated as an unknown input. Applied in radar signal processing [27], $u[i]$ is equivalent to a stochastic noise w (generally random white noise) to provide a similar effect of excitation. This implementation is called input whitening. The covariance of stochastic noise is required based on either accurate characterization of noise or online statistical learning as in [28]. Kalman filter based prediction is updated with a small time step and with high order model of Eq. (2.3) for good prediction performance. The predicted state $s(t_c + \tau_{RTT})$ can then be estimated based on recursively propagating Eq. (2.4) from t_c to $t_c + \tau_{RTT}$. However, implementing this Kalman filter based approach relies on the assumption that noise statistics can be captured accurately and requires higher computational load due to the use of a high order model.

In summary, model-free prediction methods do not need the governing equations of the vehicle dynamics or its parameters, thus are robust and independent of the remote vehicle. However, performance of existing methods depends on the validity of the assumptions on vehicle trajectory or input whitening.

2.3.1.2 Model-Based Prediction

Model-based prediction requires a full vehicle model to capture the dynamics of the remote vehicle. Various models can be used to represent vehicle dynamics with

different levels of complexity. One widely used model is the kinematic bicycle model. The kinematic bicycle model is derived based on geometric constraints, assuming zero tire slip angle. Thus, there is no lateral tire force and this assumption is reasonable at low vehicle speeds (e.g., less than 5 m/s) [66]. The equations describing this model are shown below:

$$\begin{aligned}
\dot{X} &= u \cos(\psi + \beta) \\
\dot{Y} &= u \sin(\psi + \beta) \\
\dot{\psi} &= \frac{u}{l_r} \sin(\beta) \\
\dot{u} &= a_x \\
\beta &= \tan^{-1}\left(\frac{l_r}{l_f + l_r} \tan(\delta_f)\right)
\end{aligned} \tag{2.5}$$

where vehicle parameters l_f and l_r are the distance from the center of the mass of the vehicle to the front and rear axles, respectively; X , Y , ψ describe the vehicle center of mass location and heading in the global coordinates; δ_f is the front wheel steering angle input; u is the vehicle longitudinal speed and is calculated based on the acceleration a_x . a_x can be determined based on propulsion and brake force related to control inputs of throttle and brake, as well as air drag force.

When lateral dynamics are non-negligible, especially at high vehicle speeds, the dynamic bicycle model with tire model is commonly applied [66]:

$$\begin{aligned}
\dot{u} &= vr + a_x \\
\dot{v} &= -ur + \frac{2}{m}(F_{yf} \cos \delta_f + F_{yr}) \\
\dot{r} &= \frac{2}{I_{zz}}(l_f F_{yf} - l_r F_{yr}) \\
\dot{X} &= u \cos(\psi) - v \sin(\psi) \\
\dot{Y} &= u \sin(\psi) + v \cos(\psi)
\end{aligned} \tag{2.6}$$

where r is the yaw rate. Compared to the 2 degrees-of-freedom (DoF) kinematic

model (including longitudinal and yaw) in Eq. (2.5), an additional DoF in lateral direction is added and yaw motion is determined based on moments and inertia I_{zz} . F_{yf} and F_{yr} are lateral tire forces generated by a tire model and applied to the vehicle body frame. Common tire models include the linear tire model assuming constant tire cornering stiffness C_{yf}, C_{yr} [66], the Fiala model that is related to tire-road friction coefficient [67], and the nonlinear Pacejka model that captures combined longitudinal and lateral tire slip with an empirical equation known as the Magic Tire Formula [68].

Once a suitable vehicle model is obtained, prediction of vehicle states can be implemented in two ways. One is referred to as full prediction in [20], where prediction starts with delayed vehicle states $y_2(t)$ received at t_c . The received states are the outcome of human's control commands $y_1(t_c - \tau_{RTT})$ and any updated states since then are not available yet. While waiting for the delayed states in the feedback loop, $y_2(t_c)$ is projected into future by executing the sequence of commands $y_1(t)$ (from $t_c - \tau_{RTT}$ to t_c) consecutively on the full vehicle model and the final state of the model is the predicted state $\hat{y}_2(t_c)$ at time t_c .

The second way is referred to as continuous prediction in [20]. The difference against full prediction is that prediction starts with predicted vehicle states calculated in last prediction, i.e., $\hat{y}_2(t_c - \Delta t)$. Delayed vehicle states are only used to correct the prediction to avoid potential drifting of the states between the model and the actual vehicle.

Note that model-based prediction normally outperforms model-free prediction when an accurate vehicle model is available. However, obtaining an accurate model can be difficult, as capturing the salient dynamics properly and parameterizing the model correctly are not trivial tasks. Also, when the actual vehicle to be modeled has a highly nonlinear powertrain, mapping human commands of throttle and brake to vehicle longitudinal acceleration becomes more challenging. In summary,

for model-based prediction, obtaining an accurate vehicle model is challenging and platform-dependent. Robustness may be sacrificed due to modeling errors compared to model-free prediction.

2.3.2 Display Methods

After a prediction of vehicle states is obtained, the predicted states are passed to the block of “Image Processing” and are displayed to the human operator in two main ways: frame display and full display [21].

Frame prediction display overlays the predicted state information onto the delayed view. The predicted location and heading are transformed to the local coordinates of delayed states. In [14, 69], predicted vehicle is shown as a point with an arrow showing the predicted direction. In [19, 21, 70, 71], a rectangular frame is used to represent the vehicle with width and the vehicle path from delayed position to the frame is also displayed. In [18], a semi-transparent shadow vehicle represents where the vehicle is predicted to be. The main advantage of frame display is that the computational load of overlaying predicted states as visual cue onto the delayed camera view is very low. Also, the operator can see the relative position/heading change within the duration of delays to have more insight on the vehicle movement, albeit processing this indirect information may increase the cognitive workload of the operator.

On the other hand, full prediction display modifies the delayed image from the camera and projects a camera view as if captured when the vehicle is at its predicted position with predicted heading. In [22, 72, 73, 74, 75], delayed camera view is separated into pixels or multiple regions. These pixels or regions are then warped based on the geometry of camera model and perspective transform calculated between delayed states and predicted states, and rendered together as the predicted view. In [55], the camera view of a virtual leading vehicle based on prediction is displayed to human operators in the simulated environment. As the predicted view is directly

displayed to the operators, it is similar to the original teleoperated driving setting without needing operators to become familiar with the additional overlay as in frame display. However, image rendering process is more difficult to implement and takes more time to process. Besides, after performing the transformation, the predicted view may have blank regions due to blind spots or lack of information (especially when camera needs to be rotated). For a predefined simulated environment, the above limitations are not issues, because the vehicle can be moved to the predicted location and camera view displayed is along the predicted heading direction.

A comparison between these two display strategies was performed in [21] and the conclusion was that with frame display, vehicle mobility and operator workload were similar to or slightly better than full display at low speeds, but worse at high speeds.

2.4 Prediction of the Human Operator Commands

As predictive display approach aims to predict the response of the vehicle to the operator’s commands after a round trip delay, it is also conceivable that the operator’s commands can be predicted into the future and these predicted commands are sent to the vehicle to alleviate the effect of delays. This section reviews the approaches to achieve this kind of prediction. When these approaches are implemented to compensate the whole round trip delays, there is no need to alter the display anymore, which can help reduce the computational load.

While general model-free prediction methods mentioned in Sec. 2.3.1.1 based on high order derivatives of the signals to predict could also be applied in this case, most of the methods to predict the operator commands require models to represent the complicated human driving behavior. In the literature, studying on-board driver’s acceleration, braking and steering behavior has a long history, and various driver models without considering the communication delays have been proposed [76]. However, these models may not be directly applicable to teleoperated driving with delays due

to significant difference in human operator behavior between teleoperated driving and on-board driving. When driving with delays, human operators need to estimate the vehicle outcome under their current control actions more in advance than on-board driving. With large delays, they tend to overcompensate for non-negligible error between their estimated control outcome and delayed response, resulting in control commands with oscillation induced by operators [7, 8, 9].

As far as the author is aware of, there only exists two driver models in the literature developed to simulate human’s driving behavior under teleoperation with communication delays [77, 78]. In [77], the driver model is designed for low-speed small UGVs and is in fact a PD-based controller with previewing. The received vehicle states with delays are projected ahead by an amount of lookahead time. Instant steering commands are generated using a PD controller based on the lateral error between the projected vehicle and track’s tangent line. Parameters such as lookahead time as well as the controller gains are tuned against data of human experiment to match the teleoperated driving performance. In [78], a cognitive driver model within the Adaptive Control of ThoughtRational (ACT-R) cognitive architecture can capture the steering behavior of human drivers and establish the best vehicle mobility metrics under various speed and delays. A far eyepoint is defined as a point on the track that is ahead of the vehicle by a time parameter called time headway, simulating where human drivers are staring at during the teleoperated driving. Generated steering command is proportional to the direction difference within the coordinate of camera view between current vehicle heading and the far eyepoint. Parameters of steering gains and time headway are optimally tuned for different constant vehicle speeds and various delays and the track keeping performance of the driver model are comparable to the best performance human drivers can achieve based on human-in-the-loop experiments. However, these driver models does not generate prediction of the future human commands for delay compensation.

Other works consider general applications involving human-machine interaction rather than only teleoperated driving to predict human operation into the future. A minimum jerk (MJ) model is well-known to represent the kinematics of human motions for reaching tasks (i.e., human moving arms from one point to another point) [79] and has been studied to represent human operation under the scenarios of vehicle following [80], lane change and obstacle avoidance [81]. The fundamental assumption is that human tends to generate optimized smooth motions with minimized integration of square sum of jerk (i.e., third derivative of position) over a certain period of time [79]. In other words, human motions in position can be described as 5th degree polynomials. The parameters of a single polynomial can be determined by identifying the start and end time of a minimum jerk trajectory and solving the constraint of position, velocity and acceleration at start and end time, respectively. The calculated polynomial is extrapolated afterwards to provide the predicted human operation in the future. In [26], MJ model based prediction significantly improved the teleoperation performance compared to without prediction with round trip delays of 100 ms. However, it is unknown whether the assumption that humans perform minimum jerk operation still holds under teleoperated driving with large delays.

2.5 Summary of Existing Prediction Based Methods

Prediction based methods in the literature including predictive display and prediction of operator commands are reviewed in this chapter. The former method performs prediction on vehicle states after receiving the delayed vehicle states in response to the operator's command, while the latter method predicts the operator's commands into the future and sends the predicted commands to the vehicle. All methods can be divided into model-based and model-free based on whether governing equations and parameters of the vehicle or human operator are used in the prediction. Model-based predictions lead to accurate predictions if accurate models are available, but in

practice modeling errors always exist and may sacrifice the robustness of model-based methods. Model-free prediction is robust to modeling error, but generally requires additional assumptions on the vehicle or operator motions, or on the characteristics of the noise that is assumed to drive the high order derivatives of the delayed signals. Performance of these model-free methods depends on how realistic these assumptions are or how accurately the noise statistics can be known.

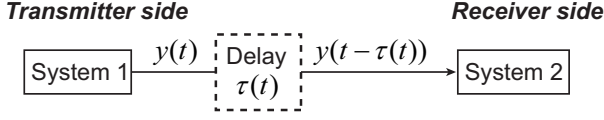
Thus, a gap in the literature is identified: A method does not yet exist that takes both performance and robustness into consideration when performing prediction on transmitted signals and compensating delays to improve mobility for high speed teleoperated UGVs. The need for such a method motivates the work described in the remainder of this dissertation.

CHAPTER III

A Novel Model-Free Predictor for Delay Compensation

In this chapter, a novel model-free predictor is developed to compensate delays. The predictor is model-free in the sense that no information about the dynamic equations governing human operators and vehicles are required to perform signal prediction and thus has the generality of predicting any signal including control commands and vehicle states as in the teleoperated vehicle system. Compared to existing model-free prediction-based methods for delay compensation summarized in II, no assumptions about the vehicle trajectory or human motion are required, nor any knowledge or estimation of noise in the high order derivatives of the transmitted signals. The predictor is also easy to implement and there is only one parameter to tune for predicting each signal of interest. Analysis of predictor stability and prediction performance are presented, and a general predictor design procedure is provided as a guideline to select the predictor parameters to ensure stability as well as pursue good prediction performance. The work in this chapter is based on publications [82, 83, 84].

(a) Setup without predictor:



(b) Setup with predictor:

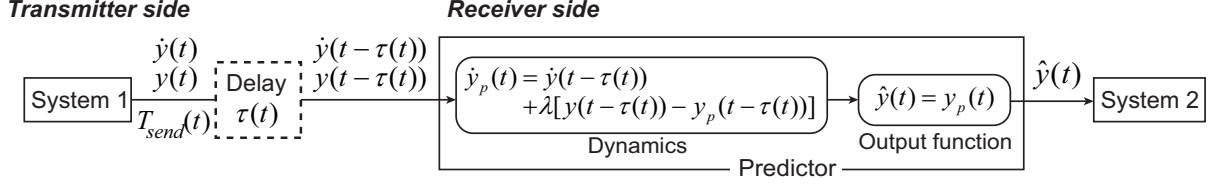


Figure 3.1: Setups of a generic one-way communication from System 1 to System 2 (a) without predictor and (b) with predictor. The novel predictor is placed at the receiver side to compensate delays and generate prediction of the original signal $y(t)$.

3.1 Predictor Dynamics

A novel predictor is developed in this work to perform prediction on a single signal. Consider a generic one-way communication from System 1 to System 2 as shown in Figure 3.1(a). The output of System 1, $y(t)$, is the signal of interest. Without the predictor, $y(t)$ is transmitted and received at the remote side with delay $\tau(t)$ and $y(t - \tau(t))$ is the input to System 2. The novel predictor is placed at the receiver side and generates a prediction based on the delayed information, as shown in Figure 3.1(b). Note that one modification is needed to use this predictor: In addition to the original transmitted signal $y(t)$, its derivative, $\dot{y}(t)$, as well as the send time stamp $T_{\text{send}}(t)$ right before transmission are included in the communication packets. The reason to include them in the packets is because the predictor developed in this work requires the information of the derivative of the signal with delays, i.e. $\dot{y}(t - \tau(t))$, and the one-way delay value $\tau(t)$ to perform the prediction. While the derivatives of some transmitted signals such as heading and position can be measured and are readily available to be directly included in the communication packets before transmission, derivative information can also be estimated through numerical differentiation with filtering to alleviate potential noise. Note that this estimation can be processed

either before sending the packets to generate $[\dot{y}(t)]_{\text{est}}$ or after receiving the delayed packets (including $y(t - \tau(t))$) at the receiver side to generate $[\dot{y}(t - \tau(t))]_{\text{est}}$ when differentiating with respect to previously received packets.

The structure of the predictor is shown in Figure 3.1(b). It is placed at the receiver side, receiving $y(t - \tau(t))$, $\dot{y}(t - \tau(t))$ and predicting $\hat{y}(t)$ to compensate delays and recover the original signal $y(t)$. The predictor dynamics is a first-order time delay system and is inspired by a model-free observer designed in [85] for Internet-distributed hardware-in-the-loop (ID-HIL) applications to alleviate the effect of delays on distorting the closed-loop dynamics and improve fidelity. In [85], the observer was developed for the case of constant delays. In this dissertation, the general case of varying delay $\tau(t)$ is considered [83] and the predictor dynamics are derived as follows.

Denote the predictor state as $y_p(t)$ and the predictor output as $\hat{y}(t)$. Here, $\hat{y}(t) = y_p(t)$ for a single signal of interest. Define the prediction error $e(t)$ between the original signal and predictor state as:

$$e(t) := y(t) - y_p(t) \quad (3.1)$$

A first-order dynamics of $e(t)$ is designed so that $e(t)$ can converge to 0 asymptotically:

$$\dot{e}(t) = -\lambda e(t) \quad (3.2)$$

where λ is a positive real parameter. Replacing $e(t)$ with (3.1), (3.2) becomes:

$$\dot{y}(t) - \dot{y}_p(t) = -\lambda(y(t) - y_p(t)) \quad (3.3)$$

Note that the derivative of the original signal, $\dot{y}(t)$ is required by the predictor, as well. However, both $y(t)$ and $\dot{y}(t)$ are transmitted from the remote site and only their

delayed signals $y(t - \tau(t))$, $\dot{y}(t - \tau(t))$ are available. Thus, (3.3) becomes:

$$\dot{y}_p(t) = \dot{y}(t - \tau(t)) + \lambda[y(t - \tau(t)) - y_p(t)] \quad (3.4)$$

where the second term on the right hand side now involves a difference between $y(t - \tau(t))$ and $y_p(t)$. That would cause the predictor to track the delayed remote signal instead of the original $y(t)$ and render the predictor obsolete. Furthermore, $y(t - \tau(t)) - y_p(t)$ cannot be meaningfully referred to as the predictor state error when states from different time instances are compared. Hence, for a more useful and meaningful definition of error, (3.4) is modified to also delay the predictor state by the same amount as $\tau(t)$, resulting in the following predictor dynamic equation:

$$\dot{y}_p(t) = \dot{y}(t - \tau(t)) + \lambda[y(t - \tau(t)) - y_p(t - \tau(t))] \quad (3.5)$$

where $y(t - \tau(t))$ and $\dot{y}(t - \tau(t))$ are inputs with delays, $y_p(t)$ is the predictor state, and $\hat{y}(t)$ is the predictor output and is the same as the predictor state, i.e. the output function of the predictor is:

$$\hat{y}(t) = y_p(t) \quad (3.6)$$

Note that the one-way delay $\tau(t)$ is included in the dynamics to generate the term of delayed predictor state $y_p(t - \tau(t))$. This delay can be determined by synchronizing the transmitter and receiver side and calculating the difference between the send time stamps in the packets and the local receive time.

There exists two main benefits of this predictor. First, the predictor is model free, i.e., no information about the dynamics or parameters of the remote system (i.e., where the delayed signals originate) is included in (3.5). Compared to other model-free prediction methods, this predictor does not require assumptions such as the vehicle moves along clothoids [20] or the signal can be extrapolated using Taylor

series expansion with estimated noise on high-order derivative terms [28]. The second benefit is the ease of implementation; the predictor dynamics is a first-order time-delay system with only one design parameter λ to predict each signal of interest. Predictor stability and performance depend on the selection of λ .

3.2 Predictor Stability Analysis

In this section, the predictor stability is studied based on the predictor error dynamics with prediction error $e(t)$ as the state. Ranges of λ to ensure a stable predictor depending on type of delays and amount of delay are provided.

Replacing (3.1) into (3.5), the following results can be derived:

$$\begin{aligned}\dot{e}(t) &= \dot{y}(t) - \dot{\hat{y}}(t) \\ &= \dot{y}(t) - \dot{y}(t - \tau(t)) - \lambda(y(t - \tau(t)) - \hat{y}(t - \tau(t))) \\ &= \dot{y}(t) - \dot{y}(t - \tau(t)) - \lambda e(t - \tau(t))\end{aligned}\tag{3.7}$$

Thus, the predictor error dynamics is defined as:

$$\dot{e}(t) = -\lambda e(t - \tau(t)) + d(t)\tag{3.8}$$

where $d(t) = \dot{y}(t) - \dot{y}(t - \tau(t))$ is considered as a disturbance input to the predictor error dynamics. Studying the stability of the predictor through error dynamics is equivalent to studying the stability of the homogeneous error dynamics with zero input, i.e. $d(t) = 0$, that is shown below:

$$\dot{e}(t) = -\lambda e(t - \tau(t))\tag{3.9}$$

The stability of the first-order delay differential system in the form of (3.9) has been well addressed in the literature. In the following sections, two methods in the

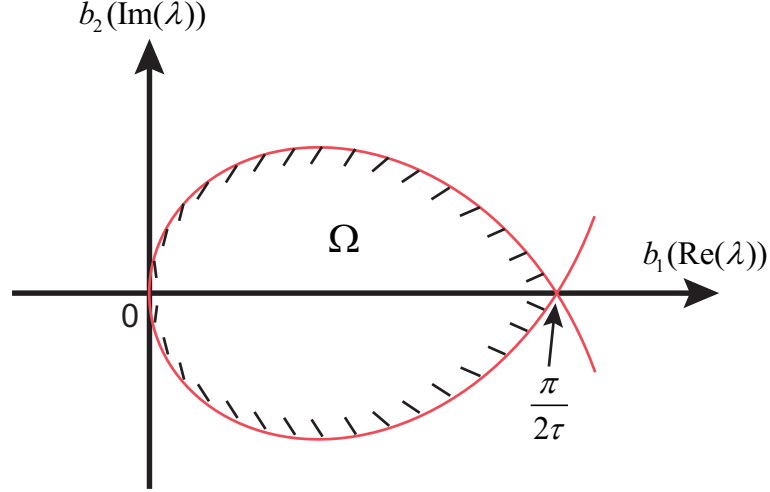


Figure 3.2: Stability region of complex λ to ensure asymptotic stability of (3.9).

literature of establishing the theoretical stability ranges for cases of constant and varying delays respectively are used. The theoretical range in the cases of varying delays is then tested against the numerical range when simulating the sinusoidal response of the predictor with actual varying communication delays.

3.2.1 Constant Delay Case

The theoretical range of λ to ensure stability of (3.9) has been given in [86, 87] and is restated here for completeness. With constant delays τ , the Laplace transform and thus the characteristic equation of (3.9) is expressed as:

$$se^{\tau s} = -\lambda \quad (3.10)$$

Denote the complex root of (3.10) as $s_r = \sigma + j\omega$ and express λ in the form of $\lambda = b_1 + jb_2$. Mori and Noldus [87] proved that all complex λ to ensure negative real part of s_r and thus asymptotic stability of (3.9) lie in an open hatched region Ω as

shown in Figure 3.2. Ω is bounded by the following curve:

$$\begin{aligned} b_1 &= \omega \sin(\tau\omega) \\ b_2 &= -\omega \cos(\tau\omega) \end{aligned} \tag{3.11}$$

where $-\frac{\pi}{2} < \tau\omega < \frac{\pi}{2}$. This boundary curve represents the values of λ that result in marginal stability and is determined by replacing $s = 0 + j\omega$ into (3.10) and balancing the real and imaginary parts on both sides. While any complex value in Ω ensures stability, the parameter λ in the predictor is defined to be real. In Figure 3.2, positive real $\lambda = b_1$ up to $\frac{\pi}{2\tau}$ is within Ω .

Therefore, the asymptotic stability of predictor error dynamics (3.8), and thus the predictor with constant delay τ , is guaranteed if and only if its parameter λ is within the following range:

$$0 < \lambda < \frac{\pi}{2\tau} = \lambda_{\max}(\tau) \tag{3.12}$$

where $\lambda_{\max}(\tau)$ is the maximum allowable λ and depends on the amount of delay τ .

3.2.2 Varying Delay Case

In the literature, the stability of (3.9) with varying delays $\tau(t)$ have been addressed with various methods. The well known $\frac{3}{2}$ stability states that under the assumption of bounded varying delay, i.e., $\tau(t) \in [0, \tau_{\max}]$, if $0 < \lambda\tau_{\max} < \frac{3}{2}$, the solution $e(t)$ is converging and thus (3.9) is asymptotically stable [88, 89]. However, this approach can be conservative for the purpose of this work, since for the hardware networks such as a wireless network there exists a small probability that the delay can exhibit a sharp increase [90], thereby causing a large τ_{\max} and thus a λ that is unnecessarily small. In addition, due to network mobility, packet routing and signal interferences, communication delays are hard to bound exactly [91, 92].

Alternative methods exist based on the Lyapunov-Krasovskii functional or Lyapunov-

Razumikhin functions [93, 94, 95, 96, 97]. In these works, a more general retarded delay differential system in (3.13) is studied and the bounds of parameters a and λ to ensure stability are mostly determined via linear matrix inequalities (LMIs).

$$\dot{e}(t) = ae(t) - \lambda e(t - \tau(t)) \quad (3.13)$$

When $a = 0$, the bounds of λ can be used as the stability range of the predictor error dynamics. However, as far as the author is aware, these methods have a more conservative bound of the maximum allowable delay than the $\frac{3}{2}$ stability bound.

While the above methods consider delays at each time instance as a discrete value and this value changes continuously to represent varying delays $\tau(t)$ as a function of time, some researchers consider delays to be varying in a deterministic way for analysis purposes. For example, relating a fast varying discrete delay with a distributed delay, a comparison system can be used along with a frequency domain method to analyze and derive the stability criteria [98]. Other researchers consider random delays to follow a Markov Chain model and prove the stability of linear time delay systems [99, 100, 101].

In this dissertation, the approach of studying robust stability with random delays that are modeled by a two-time-scale Markov chain is leveraged [102, 103]. The theorem is summarized below for completeness and an open-loop test is performed to verify that the theorem is applicable to the types of networks that are of interest for this work.

In [103], varying delays $\tau(t)$ are modeled as a continuous-time Markov chain of delays in a finite state space $\mathbb{S} = \{\tau_1, \tau_2, \dots, \tau_m\}$, $m \in \mathbb{N}$. \mathbb{S} is chosen based on the distribution of the potential delays and the desired accuracy. For example, if the network delays are in the order of tens of ms and vary within 1 s, a suitable selection can be $\tau_k = 10k \ \forall k \in \{1, 2, \dots, 100\}$ in ms ($m = 100$). In practice, one can always

choose a very large \mathbb{S} so that a large or dense range of delay values is covered.

Assume the Markov chain is composed of fast and slow motions. Slow motions refer to the property that the network is steady most of the time and delay is changing slowly, while fast motions denote the sudden spikes in the network due to network traffic or different packet routes and thus delay is highly variable. To reflect these spikes, a small parameter $\varepsilon > 0$ is introduced so that the Markov chain contains two time scales, a usual running time t and a stretched time t/ε . The Markov chain is modeled using the generator:

$$Q^\varepsilon = \frac{Q}{\varepsilon} + Q_0, \quad (3.14)$$

where $\frac{Q}{\varepsilon}$ and Q_0 represent the fast-varying and slow-varying parts, respectively.

Thus, (3.9) becomes a random switching system among m fixed delay subsystems:

$$\dot{e}(t) = -\lambda e(t - \tau_k) \quad k = 1, 2, \dots, m. \quad (3.15)$$

The results are based on the average delay τ_{avg} , which is defined with respect to the stationary measure v_k :

$$\tau_{\text{avg}} = \sum_{k=1}^m \tau_k v_k \quad (3.16)$$

Theorem 3.2 in [103] states that if all delay states $\tau_1, \tau_2, \dots, \tau_m \geq \frac{1}{\lambda}$ and $0 < \lambda \tau_{\text{avg}} < \frac{3}{2}$, then the trivial solution of the homogeneous version of (3.15) is almost surely asymptotically uniformly stable as $\varepsilon \rightarrow 0$, under the required assumption that delay sequence can be modeled using a two-time-scale Markov Chain.

To test the utility of this theorem for the purposes of this work with actual communication delays, network delays were measured between Ann Arbor, USA and Shanghai, China (AA-SH). Note that delay sequence in this Internet based network has the representative characteristics as in the common commercial networks such as wired/wireless communication, where delays are mostly varying around their mean value and certain spikes in delays are observed due to the instantaneous heavy load

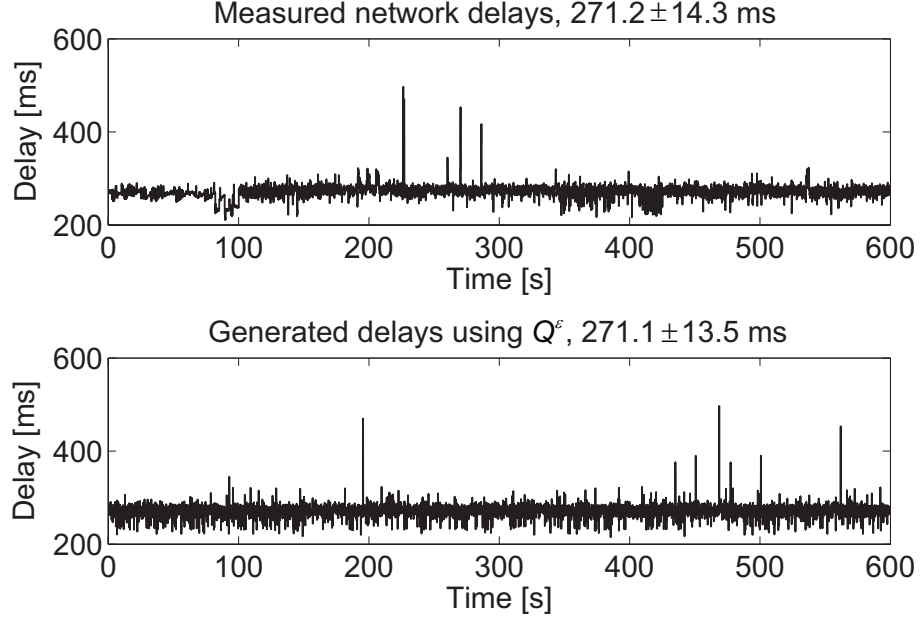


Figure 3.3: Above is the round-trip delay measurement between Ann Arbor, USA and Shanghai, China (AA-SH). Below is the generated delay sequence using the two-time-scale Markov Chain. Mean and standard deviation of the delays between the measured data and generated delay sequence are close to each other.

within the network traffic. Additional measurement of delays and stability boundary test are needed for some other representative networks commonly used in for teleoperated UGVs, such as Satellite network and cell phone network. Using the ping test in the Windows command console, packets with size of 1 KB were sent every 0.1 s and the round trip delays were measured. The upper graph in Figure 3.3 shows the delay data measured in a 10-minute window. The drop rate is 4.6 % and the mean delay is 271.2 ms with standard deviation of 14.3 ms.

First, the assumption of using a two-time-scale Markov chain to model network delays is verified. Because the network delays are measured for a long time, delays are considered to have reached the stationary measure v_k already. The average delay τ_{avg} is calculated based on (3.16). The generator matrix Q^ε is found via empirical transition matrices [104]. After checking that a potential Q^ε exists based on the

theorems in Section 2 and 3 in [104], Q^ε is calculated according to

$$Q^\varepsilon = \frac{(P - I) - \frac{(P-I)^2}{2} + \frac{(P-I)^3}{3} - \frac{(P-I)^4}{4} + \dots}{dt} \quad (3.17)$$

where the transition matrix P is determined based on the sequence of the measured delays, I is the identity matrix that is of the same size as P , and dt is the time step between transitions of delays. To ensure that off-diagonal entries of Q^ε are non-negative for a proper Markov chain, the negative entries are set to be 0 and the sum of them are added back to the corresponding diagonal entry to preserve the property of the generator matrix to have row-sums of 0 [105]. The final generator Q^ε is thus determined. The diagonal terms are on the order of 1 and are significantly greater than the off-diagonal terms. Setting $\varepsilon = 0.1$ and using two constant Markov generators Q and Q_0 with all terms between -1 and 1, $\frac{Q}{\varepsilon}$ and Q_0 can represent the diagonal and off-diagonal terms in Q^ε . Thus, the generator Q^ε to model the random delay sequence can be expressed in two time scales, as in (3.14).

Using this constant generator matrix Q^ε , a two-time-scale Markov chain model of the delay sequence is generated. Let the first delay to be the same as the first delay measured and initial transition matrix to be $P^\varepsilon(0) = I$. Note that $P^\varepsilon(t)$ is changing based on the dynamics:

$$\frac{dP^\varepsilon(t)}{dt} = P^\varepsilon(t) \cdot Q^\varepsilon \quad (3.18)$$

The lower graph in Figure 3.3 shows one of the generated delay sequences for the same amount of time as the measured delays, with time step of 0.1 s. The mean and standard deviation are 271.1 and 13.5 ms, respectively, which are very close to those of the measured delays. Figure 3.4 shows the histogram of delay distribution for the measured and generated delay sequences. The distributions of delays are close to each other, as well. Thus, the assumption that the random delays follow a two-time-scale Markov chain model with the generator in (3.14) is found to be suitable for the

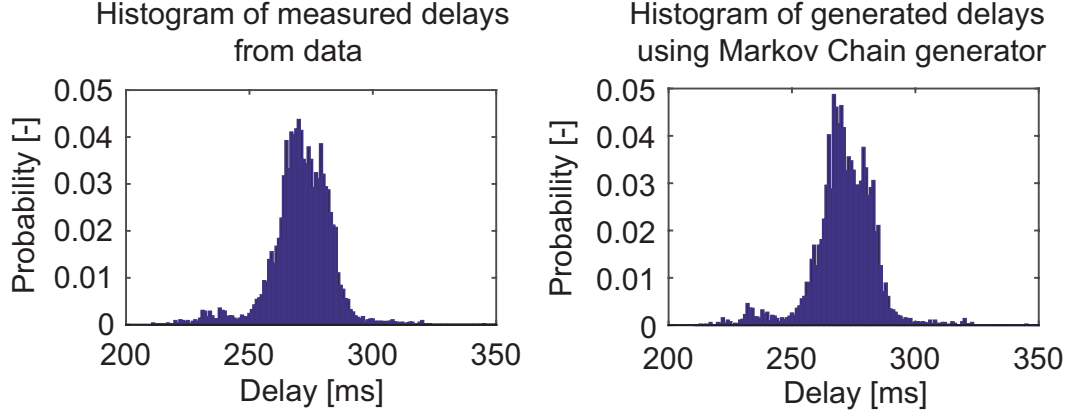


Figure 3.4: Above and below are the histograms of the measured and generated delay sequences. The distributions of delays are close to each other.

example network that is considered in this work.

Next, the actual stability boundary of the predictor is compared against the theorem for four different networks to test the utility of the theoretical results. Recall the setup with predictor in Figure 3.1(b) and assume that the signal of interest is sinusoidal with frequency of 0.1 rad, i.e. $y(t) = \sin(0.1t)$. Three different networks along with the AA-SH network mentioned above are employed to capture various varying delays $\tau(t)$. The three network delay data have been collected in [45] and their histograms are shown in Figure 3.5, with mean round-trip delays ranging from 25.5 ms to 116.2 ms. Based on these measured delays, it is verified using the procedure described above that the two-time-scale Markov chain assumption is satisfied and a new delay sequence is generated using the two-time-scale Markov chain model for each network.

Figure 3.6 shows the stability ranges for the delay sequences generated by the Markov chains, where the range is characterized by the maximum allowable predictor parameter value λ to ensure that the predicted output is bounded and stable. For all four networks, the actual ranges are found numerically via simulation of the predictor. The theoretical ranges given by the $\frac{3}{2}$ stability theorem and the two-time-scale Markov

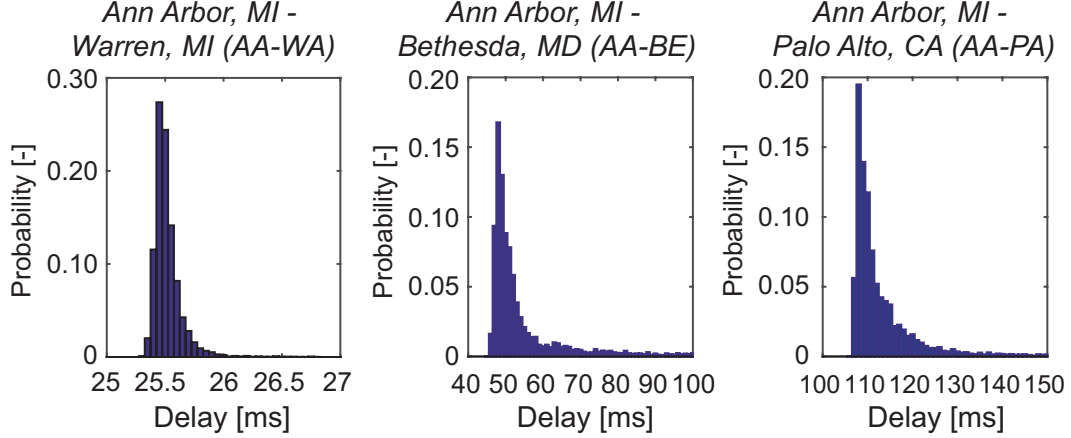


Figure 3.5: The histogram of round-trip delay measurements from three different networks.

chain approach are $0 < \lambda < \frac{3}{2\tau_{\max}}$ and $0 < \lambda < \frac{3}{2\tau_{\text{avg}}}$, respectively. It is noted that the two-time-scale Markov chain approach provides a sufficient, yet not very conservative range of the parameter λ to ensure a stable predictor compared to the numerical results. The range based on the $\frac{3}{2}$ stability theorem is more conservative than the one from the Markov chain approach, especially when delays vary in a large range and $\tau_{\max} \gg \tau_{\text{avg}}$, like in the case of AA-BE network. Another potential benefit of the Markov chain approach could be that if the network is assumed to be stationary within a certain time window, stability range may change with respect to the moving average delays $\tau_{\text{avg}}(t)$ instead of the average values of delays in long history to represent more recent conditions in the network. In fact, this idea of using predictor stability range with respect to the moving average delay is applied in the later case study when dealing with a hardware network in Section 3.4.3.

In conclusion, predictor stability with varying delays is established when the network delays meet the assumption that they can be captured with a two-time-scale Markov chain. Predictor is asymptotically stable under such assumption if its param-

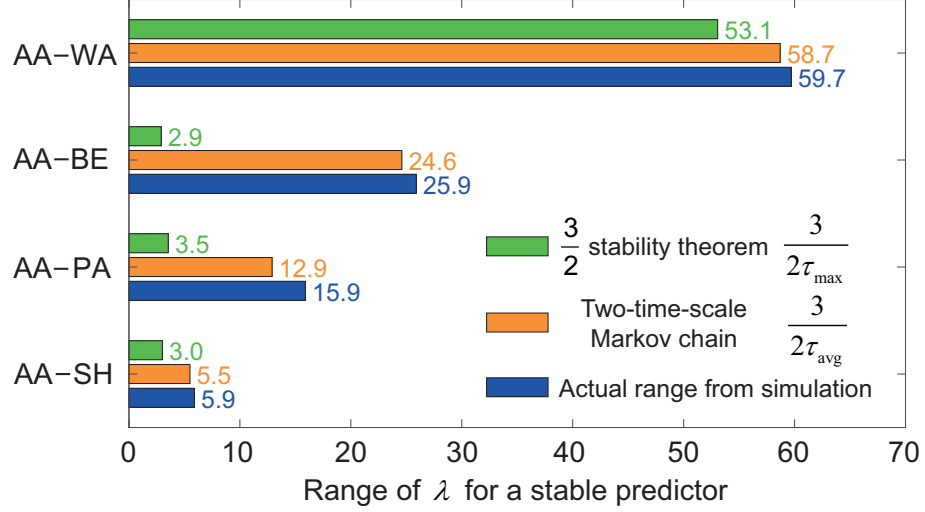


Figure 3.6: Range of predictor parameter λ to ensure a stable predictor. The theorem based on the two-time-scale Markov chain provides a sufficient, yet not very conservative, range of the parameter λ .

eter λ is within the following range:

$$0 < \lambda < \frac{3}{2\tau_{\text{avg}}(t)} = \lambda_{\text{max}}(\tau(t)) \quad (3.19)$$

where $\lambda_{\text{max}}(\tau)(t)$ is the maximum allowable λ for varying delays $\tau(t)$ and $\tau_{\text{avg}}(t)$ is the moving average of delays within a period of time.

In summary, the predictor stability criterion is concluded as follows: The predictor is asymptotically stable when λ is chosen within the following ranges [83]:

$$\begin{aligned}
 0 < \lambda < \frac{\pi}{2\tau} &= \lambda_{\text{max}}(\tau) && \text{for constant delays } \tau \\
 0 < \lambda < \frac{3}{2\tau_{\text{avg}}(t)} &= \lambda_{\text{max}}(\tau_{\text{avg}}(t)) && \text{for varying delays } \tau(t) \text{ that can be modeled} \\
 &&& \text{in a two-time-scale Markov Chain}
 \end{aligned} \quad (3.20)$$

Note that when the moving average $\tau_{\text{avg}}(t)$ is the same as the constant delay value τ , the maximum allowable λ for varying delays is slightly smaller than that for constant delays. Predictor stability with varying delays is discussed as almost surely

asymptotically stability.

3.3 Frequency Domain based Performance Analysis

While the range of λ to ensure a stable predictor is established for the cases of both constant and varying delays, it does not provide any insight into the performance of the predictor. In this section, a predictor performance analysis for constant delays is presented in the frequency domain. The predictors are then implemented and integrated on a simple motor-shaft-torque networked closed-loop system as a case study to test the predictor performance against the analysis with various λ and also evaluate the closed-loop performance with the addition of predictors.

With constant delays τ , in Figure 3.1, define the coupling error $c(t)$ between the original and delayed signal:

$$c(t) := y(t) - y(t - \tau) \quad (3.21)$$

$c(t)$ quantifies how much disturbance delays cause on the original signal. Recall the prediction error $e(t) = y(t) - y_p(t) = y(t) - \hat{y}(t)$ defined in (3.1), which represents how close the prediction is to the original signal. Ideally, $e(t)$ should be zero to completely eliminate the delay effect. Hence, a good choice of the design parameter λ should not only guarantee the predictor's stability, but also minimize $e(t)$ for better prediction performance. In this work, predictor performance is characterized by the extent to which the predictor output $\hat{y}(t)$ is more similar to the original signal $y(t)$ than the delayed signal $y(t - \tau)$. In other words, the relative magnitude of $c(t)$ and $e(t)$ are compared. A frequency domain analysis is thus performed to study how selection of λ affects the relationship between $c(t)$ and $e(t)$ over various frequencies of $c(t)$ [106, 82]. The analysis is presented as follows.

With constant delays, the error dynamics in (3.8) can be rewritten in the following

form:

$$\dot{e}(t) = -\lambda e(t - \tau) + \dot{c}(t) \quad (3.22)$$

The following transfer function from the input $c(t)$ to the output $e(t)$ is then obtained from (3.22):

$$\frac{E(s)}{C(s)} = \frac{s}{s + \lambda e^{-\tau s}} \quad (3.23)$$

The gain of the transfer function (3.23) is denoted as $M(\omega)$:

$$M(\omega) = \frac{|\omega|}{\sqrt{\omega^2 - 2\omega\lambda \sin(\tau\omega) + \lambda^2}} \quad (3.24)$$

$M(\omega)$ represents the ratio of the prediction error to the coupling error in steady state at a given frequency ω and depends on λ and τ . When $M(\omega) < 1$, predictor attenuates the coupling error $c(t)$ and improves performance, whereas when $M(\omega) > 1$, then the coupling error at that frequency is amplified, which could lead to a bad predictor performance even if the predictor is stable.

$M(\omega)$ for various values of τ and λ are shown in Figure 3.7 and Figure 3.8. Note that $\lambda_{\max}(0.03)$ refers to the maximum allowable λ for a stable predictor with delays of 0.03 s. In Figure 3.7, delays are fixed to be 0.03 s and $M(\omega)$ with λ equal to different scales (0.15 - 0.90) of $\lambda_{\max}(0.03)$ are plotted, while in Figure 3.8, $\lambda = 7.9 = 0.15\lambda_{\max}(0.03)$ is fixed and $M(\omega)$ with delays from 0.03 s to 0.135 s are compared. All $\{\lambda, \tau\}$ pairs shown satisfy the stability criterion (3.20). A number of observations can be made in these figures regarding the performance characteristics of the predictor. First, Figure 3.7 illustrates that the steady state performance is better at low frequencies for larger λ values. However, it is not always the case that larger λ gives better steady state tracking. Namely, within the range of about 30 rad/s to 200 rad/s, small λ values may be a better choice for this example delay value, since larger λ values lead to an overshoot above 0 dB in the magnitude plot. Furthermore, at higher frequencies, the predictor is less effective in terms of attenuating the state

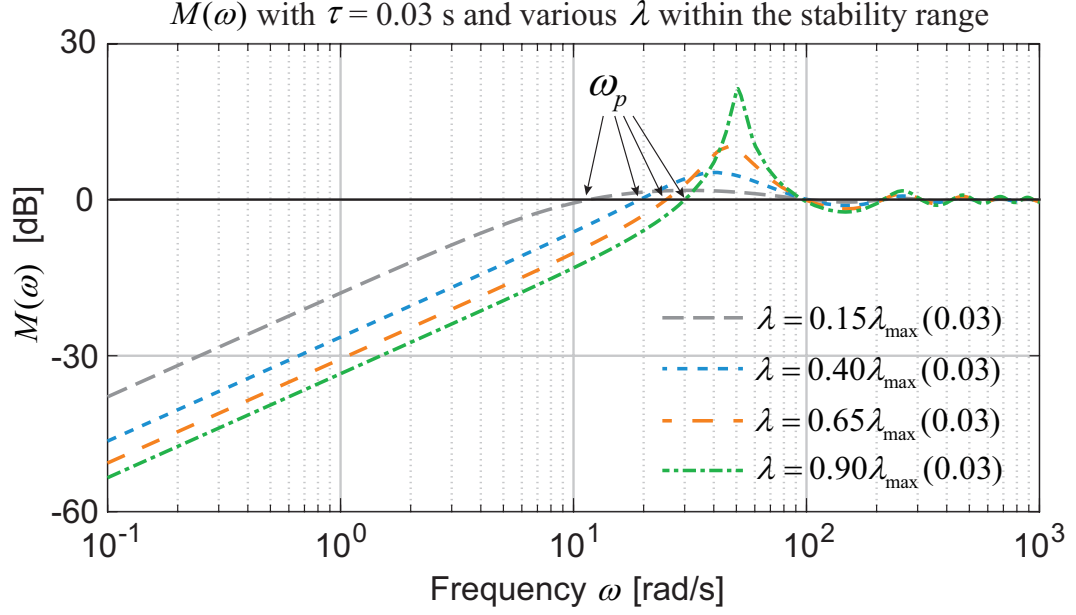


Figure 3.7: Gain of (3.23), $M(\omega)$, with fixed $\tau = 0.03$ s for various λ values within the stability range. Predictor bandwidth ω_p increases with larger λ .

tracking error, since the magnitude remains close to 0 dB regardless of the λ value chosen, i.e., $M(\omega)$ is approximately one for large ω values. Finally, Figure 3.8 shows that it is more difficult for the proposed predictor to be effective as the delay τ becomes larger, since the frequency range corresponding to a $M(\omega)$ smaller than 0 dB becomes smaller as the delay increases.

Here, a property of the predictor called the predictor bandwidth ω_p is introduced [84]. It is defined as the smallest ω satisfying $M(\omega) = 1$. ω_p is related to λ based on (3.24):

$$\lambda = 2\omega_p \sin(\tau\omega_p) \quad (3.25)$$

With a given constant delay τ , considering both (3.20) and (3.25) to ensure a stable predictor, we have $0 < \tau\omega_p \sin(\tau\omega_p) < \pi/4$. By solving it numerically, ω_p is found in the range:

$$0 < \omega_p < \frac{0.959}{\tau} \quad (3.26)$$

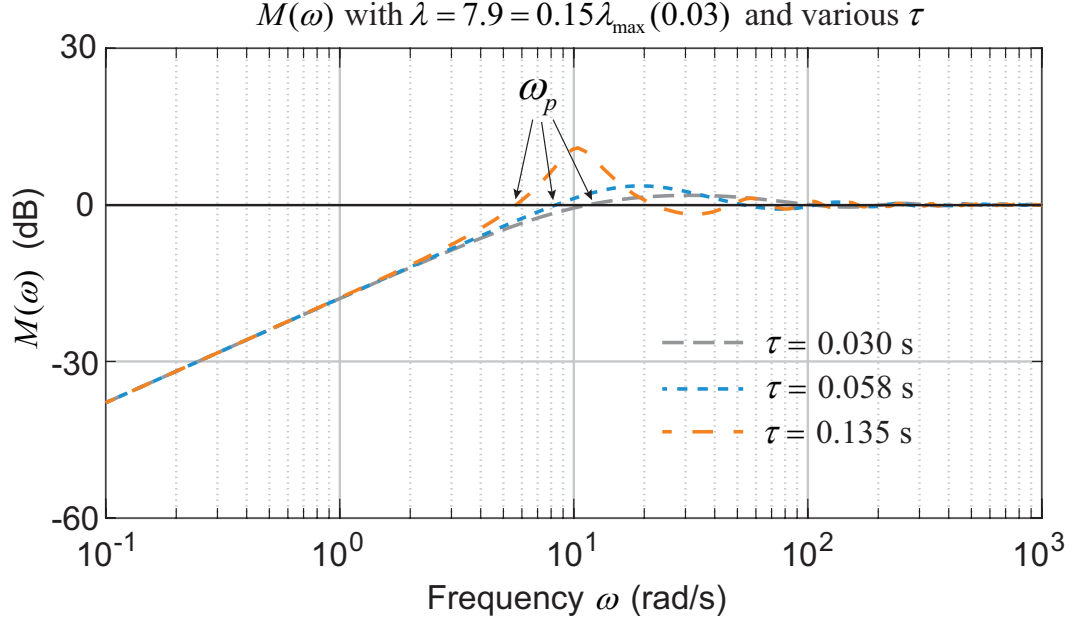


Figure 3.8: Gain of (3.23), $M(\omega)$, with fixed $\lambda = 7.9 = 0.15\lambda_{\max}(0.03)$ for various constant delay values. Predictor bandwidth ω_p increases with smaller τ .

Based on expression of (3.25) and (3.26), a larger bandwidth ω_p is expected in the conditions of larger λ and smaller τ . The same trend is observed in Figure 3.7 and Figure 3.8, where ω_p for different pairs of $\{\lambda, \tau\}$ are marked.

Note that ω_p is a critical value to distinguish the trend of predictor steady-state performance with respect to different frequencies. Specifically, predictor is always effective in improving performance within the predictor bandwidth, i.e., $\omega \in [0, \omega_p)$, but could significantly worsen the performance when ω is beyond ω_p , albeit the predictor will not have any impact on the performance in the limit as $\omega \rightarrow \infty$. Thus, when designing the predictor, ω_p is an important property to consider.

In summary, (3.23) establishes the relationship between the steady-state performance of the predictor, its design parameter λ , and the time delay τ . Hence, this analysis is important to understand some fundamental, frequency-domain steady state performance characteristics of the predictor, e.g. the predictor bandwidth ω_p . It is still unknown, however, how the predictor will perform in transient when introduced

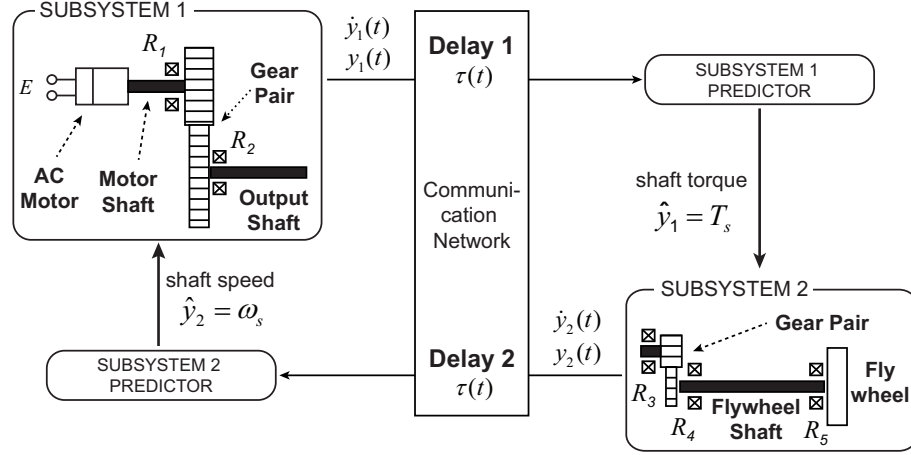


Figure 3.9: Networked motor-shaft-gear system with bilateral communication delays $\tau(t)$. Coupling signals are the shaft torque T_s and the shaft speed ω_s .

into a closed-loop system. This motivates the case study in the next section.

3.4 Case Study

This section presents a case study to demonstrate the performance of predictors in terms of reducing the negative impact of communication delays on the fidelity of the networked closed-loop integration of distributed systems [83, 82].

Note that because this model-free predictor does not require information about the dynamics of the system where the signals originate, it can be integrated with any linear or nonlinear system in general to perform prediction on signals. Before implementing predictors on the nonlinear closed-loop vehicle teleoperation system in Figure 1.1, which requires human operators in the loop, a linear example of networked motor-shaft-gear system is considered first to provide some insight of the prediction accuracy of the predictors, as well as their effect on the closed-loop performance.

The example system is shown in Figure 3.9 and contains two distributed subsystems that communicate with each other. In Subsystem 1, an AC motor with voltage input E drives the output shaft through a gear pair and outputs the shaft torque.

Subsystem 2 is located remotely from Subsystem 1. It takes the shaft torque T_s as the input through the network. With a flywheel driven through a gear pair as load, the output of the subsystem is the rotational shaft speed ω_s on the same shaft. Internal resistance of the motor and rotary bearings on all the shafts are considered, along with the compliance of the output shaft. Thus, the coupling signals that are communicated over the network are the shaft speed, ω_s , from Subsystem 2 to Subsystem 1 and the shaft torque, T_s , from Subsystem 1 to Subsystem 2. Assume the one-way delays $\tau(t)$ between Subsystem 1 and Subsystem 2 are the same and are set to be half of the round-trip delays measured. The networked system without the predictors can tolerate constant one-way delays τ of up to 175 ms to remain input-output stable. Lumping all the subsystem parameters together, the dynamics of each subsystem after simplification are given as:

Subsystem 1:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 500 \\ -0.0055 & -0.53 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 & -1 \\ 0.00026 & 0 \end{bmatrix} \begin{bmatrix} E \\ \omega_s \end{bmatrix} \quad (3.27)$$

$$y_1 = T_s = 0.28x_1$$

Subsystem 2:

$$\begin{aligned} \dot{x}_3 &= -0.1x_3 + 0.44T_s \\ y_2 &= \omega_s = 10x_3 \end{aligned} \quad (3.28)$$

The Bode plots of Subsystem 1 and Subsystem 2 are shown in Figure 3.10. Both subsystems act like low-pass filters. The cutoff frequency of Subsystem 1 for external voltage input E is 2.5 rad/s.

Two predictors developed in this work are added as in Figure 3.9. The dynamic equations of the two predictors are:

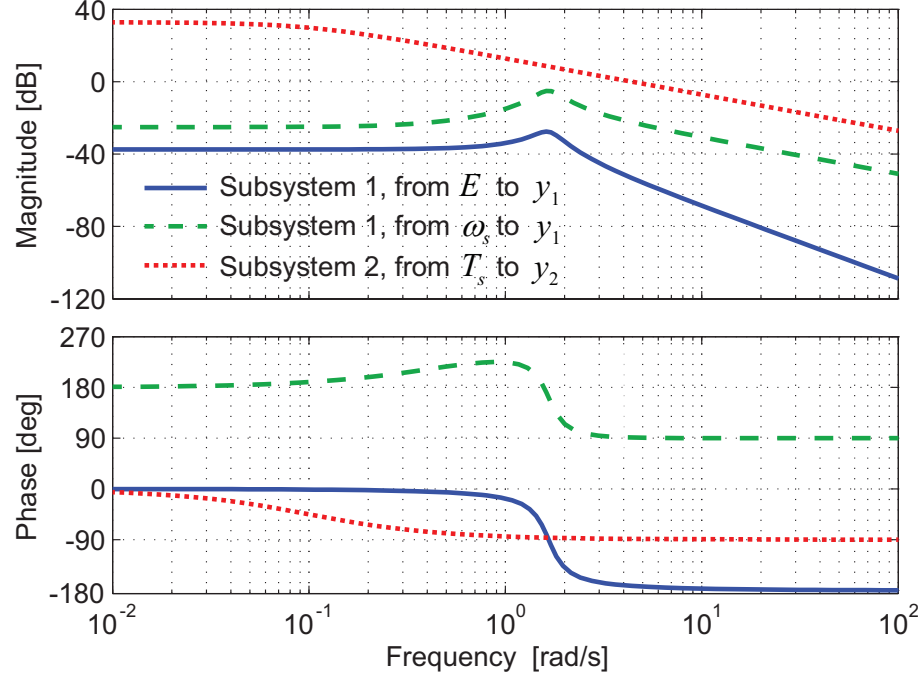


Figure 3.10: Bode plot of Subsystem 1 (from E to y_1 and from ω_s to y_1) and Subsystem 2 (from T_s to y_2). Subsystem 1 has cutoff frequency of 2.5 rad/s for input E .

Subsystem 1 predictor:

$$\begin{aligned}\dot{y}_{p1}(t) &= \dot{y}_1(t - \tau(t)) + \lambda_1 [y_1(t - \tau(t)) - y_{p1}(t - \tau(t))] \\ \hat{y}_1(t) &= y_{p1}(t)\end{aligned}\tag{3.29}$$

Subsystem 2 predictor:

$$\begin{aligned}\dot{y}_{p2}(t) &= \dot{y}_2(t - \tau(t)) + \lambda_2 [y_2(t - \tau(t)) - y_{p2}(t - \tau(t))] \\ \hat{y}_2(t) &= y_{p2}(t)\end{aligned}\tag{3.30}$$

Note that the parameters λ_1 and λ_2 used in the predictors are selected to be the same for simplicity. In general, different values within the stability region can be chosen. The prediction accuracy of the predictor is evaluated by comparing prediction error $e(t)$ with respect to coupling error $c(t)$.

The closed-loop performance is studied based on the final output of the networked

system under the same driven voltage input. In this example, the final output is the difference in angular position between the two ends of the output shaft in Subsystem 1, $\Delta\theta = x_1$. Three cases are tested: the ideal case without delays, the delayed case without the predictors, and the predictor case with two predictors to compensate the delays. Consider the simulation output of the ideal case as the benchmark. Three metrics are introduced to help with the comparison. The 2-norm of the difference between the outputs of the delayed and ideal cases is denoted as p_0 , whereas p is similarly defined between the predictor and ideal cases. To quantify the closed-loop performance with predictors compared to the delayed case, a normalized performance metric p_n is used. Mathematically, these metrics are expressed as below:

$$p_0 = \|\Theta_d - \Theta_i\|_2, \quad p = \|\Theta_p - \Theta_i\|_2, \quad p_n = \frac{p}{p_0} \quad (3.31)$$

where $\Theta = \Delta\theta$ in rad is the output trajectory vector with subscripts i, d, p representing the ideal, delayed and predictor cases, respectively. Smaller p and p_n indicate a better compensation of the delays by the predictors. When $p = p_n = 0$, the output of the predictor case is identical to that of the ideal case and the delays are fully compensated. $p_n > 1$ means that the predictors worsen the fidelity of the networked integration compared to the delayed case, and $p_n < 1$ means that the predictors improve the fidelity. Three parts of the simulation results to evaluate the prediction accuracy of the predictors and closed-loop performance with predictors are shown below.

3.4.1 Performance versus Input Frequencies

In the first part of the case study, performance is studied when signals of different frequencies are predicted by the predictors.

Consider one-way constant delays $\tau = 0.03$. From Figure 3.7, the steady-state

performance is improved when the frequency of $c(t)$ is less than 10 rad/s for all λ in the stability range and becomes worse when ω is around 30-100 rad/s. Beyond 100 rad/s, there is no improvement or degradation in steady state performance. To verify the observation, sinusoidal voltage inputs $E = 12 \sin(\omega t)$ volts with various excitation frequency ω of interest from 0.5 to 100 rad/s are applied to the Subsystem 1 and the simulation is run for 30 s at a fixed time step of 0.005 s with zero initial conditions on all subsystems and predictor states. Different λ values ensuring stable predictors is tested as well to find out the λ for maximum performance improvement.

At each ω , performance metrics p_0 without predictors and p_n with predictors using various λ values are summarized in Table 3.1. $\lambda = 0$ corresponds to the case when predictors are not used and p_0 represents how distorted the output signal is due to delays. Note that p_0 is relatively small at low $\omega = 0.5$ rad/s, because delays do not cause much negative effect for low frequency signals and are very small at high $\omega \geq 10$ rad/s, because according to the system Bode plot in Figure 3.10, both Subsystems act like low pass filters and significantly attenuate the input magnitude at high frequencies. Despite the small p_0 , predictors with all λ tested are capable of improving the closed-loop performance (i.e. $p_n < 1$) for all ω tested.

The level of improvement is similar when dealing with inputs of high ω and low ω . One explanation is that both subsystems exhibit a low-pass filter type behavior and attenuate the higher frequency signals. In general, a larger λ value corresponds to better predictor performance in terms of attenuating the effect of delay as can be seen by the smaller p_n values. Two specific ω values will be discussed further to aid with the comparison between the frequency-domain analysis of Section 3.3 and prediction accuracy based on the time-domain simulation results.

When ω is 1.5 rad/s, $M(\omega) < 1$ based on Figure 3.7, and the predictor is expected to attenuate coupling error $c(t)$. The prediction error $e(t)$ of the signal $y_2(t)$ (i.e. shaft speed ω_s) for the Subsystem 2 Predictor is calculated based on simulation results and

Table 3.1: Performance metrics for different λ values and excitation frequencies over a simulation time window of 30 s. A smaller metric value indicates better performance.

	ω (rad/s)	0.5	1.5	10	30	50	100
p_0	$\lambda = 0$	0.39	3.63	0.21	0.07	0.04	0.02
p_n	$\lambda = 0.15\lambda_{\max}(0.03)$	15.7%	19.0%	22.1%	21.8%	21.8%	21.8%
	$\lambda = 0.40\lambda_{\max}(0.03)$	5.9%	7.2%	8.4%	8.2%	8.2%	8.2%
	$\lambda = 0.65\lambda_{\max}(0.03)$	3.7%	4.4%	5.1%	5.1%	5.1%	5.0%
	$\lambda = 0.90\lambda_{\max}(0.03)$	2.6%	3.2%	3.7%	3.7%	9.1%	3.6%

is shown in Figure 3.11 for two λ values. Note that for different λ values, $c(t)$ may be different, because adding predictors modifies the closed-loop system. Figure 3.11 shows that with the larger λ value, the predictor gives a faster response to reach steady state, and the magnitude of $e(t)$ is smaller than that of $c(t)$ in steady state, which is consistent with the frequency-domain analysis in Section 3.3.

However, the same analysis shows that at larger frequencies, the predictors amplify $c(t)$ at steady state. For example, When ω is 50 rad/s, all λ values result in amplification of $c(t)$ and larger λ in this condition causes much more amplification. The simulation results of $e(t)$ with $0.65\lambda_{\max}(0.03)$ and $0.90\lambda_{\max}(0.03)$ shown in Figure confirm the degradation in prediction accuracy at steady state. With larger λ , significant amplification of steady state $c(t)$ results in less improvement in the closed-loop performance (9.1% with larger λ compared to 5.1% with smaller λ). However, the closed-loop performance is still improved when the predictors with these two λ are applied. The reason why the predictors are still effective in terms of p_n is that p_n captures the transient response, as well. The transient response does not only include the excitation frequency itself, but also the lower frequencies, where the predictors are effective.

The outputs from ideal case, delayed case and predictor case with $\lambda = 0.15\lambda_{\max}(0.03)$ and $\lambda = 0.90\lambda_{\max}(0.03)$ are compared in Figure 3.13, when ω is 50 rad/s. The outputs in the predictor case are much closer to the output in the ideal case, indicating that predictors are still capable of improving the closed-loop performance under high

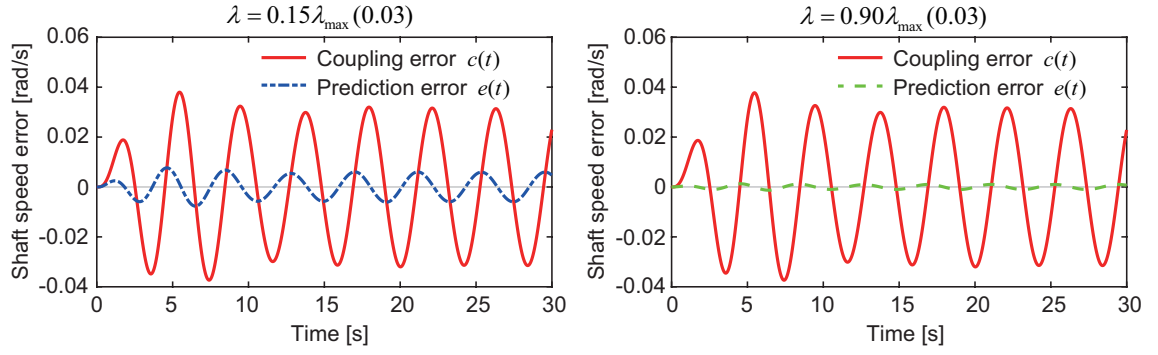


Figure 3.11: Subsystem 2 predictor performance for $\omega = 1.5$ rad/s

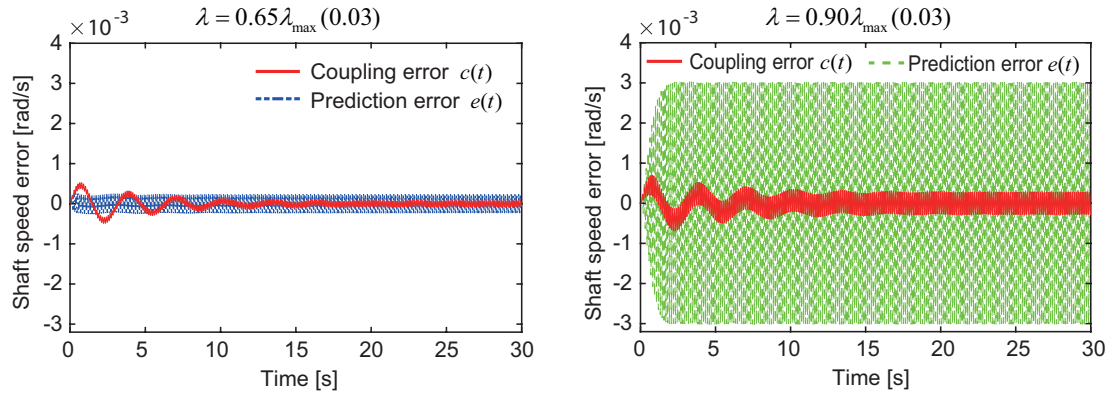


Figure 3.12: Subsystem 2 predictor performance for $\omega = 50$ rad/s

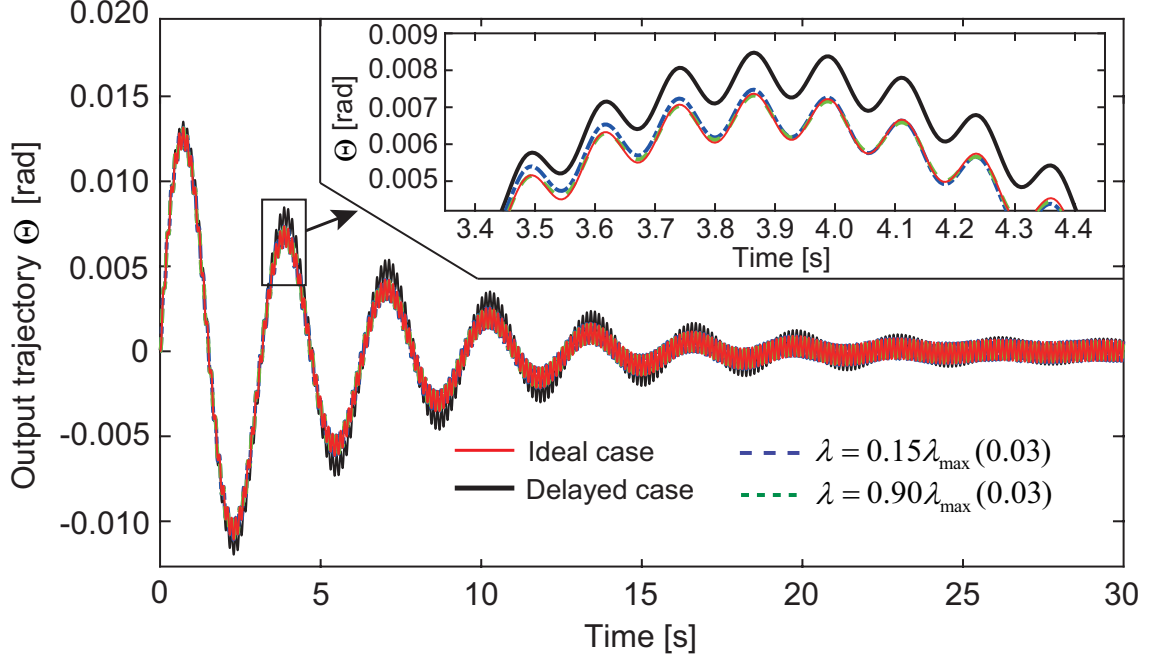


Figure 3.13: Output trajectory for $\omega = 50$ rad/s. Predictors are capable of improving closed-loop performance under high frequency input.

frequency input, despite the amplified prediction errors at steady state.

In conclusion, simulation results in this section validate the predictor performance analysis in Section 3.3 and show the potential of predictors in improving the closed-loop performance subject to inputs with various frequencies.

3.4.2 Performance with Simulated Delays

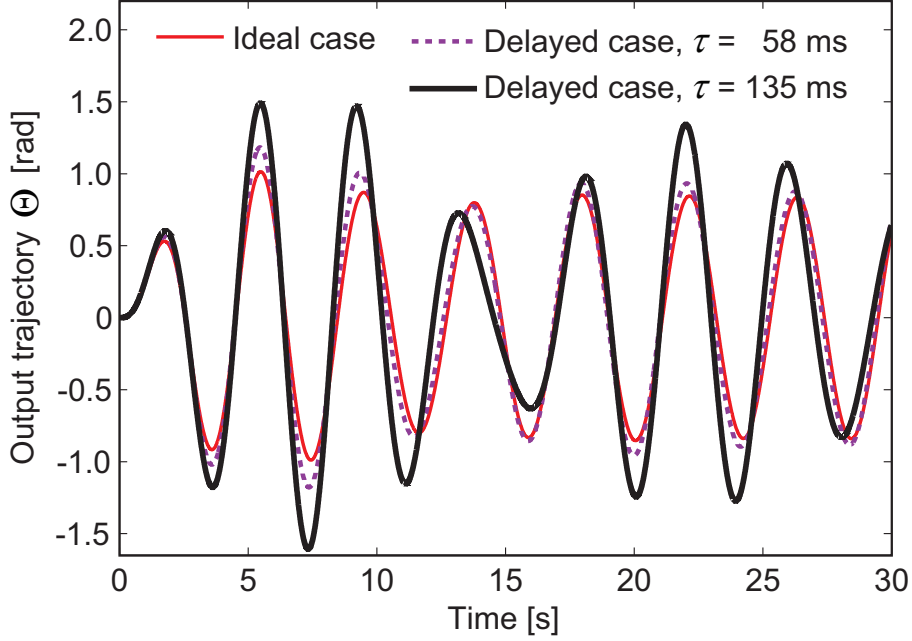
In the second part of the case study, the example system is run using various simulated constant and varying delays. The closed-loop performance under constant and varying delays is compared to each other.

3.4.2.1 Constant Delays

Three constant one-way delays τ are tested: 30 ms, 58 ms and 135 ms. For each delay, based on (3.12), the maximum allowable parameter that ensures a stable predictor is attained as $\lambda_{\max} = \frac{\pi}{2\tau}$. Then, four λ values are selected as 15%, 40%,

Table 3.2: Performance metrics with different constant delays

Metrics \ Delay		One-way Constant Delay τ		
		30 ms	58 ms	135 ms
p_0 (rad)		3.63	7.61	26.31
p (rad)	$\lambda = 0.15\lambda_{\max}(\tau)$	0.69	2.53	16.23
	$\lambda = 0.40\lambda_{\max}(\tau)$	0.26	0.98	5.46
	$\lambda = 0.65\lambda_{\max}(\tau)$	0.16	0.60	3.32
	$\lambda = 0.90\lambda_{\max}(\tau)$	0.12	0.44	2.38

**Figure 3.14:** Comparison of the output trajectory between ideal case and delayed cases with different delays. Larger delay has worse impact on the system.

65%, 90% of $\lambda_{\max}(\tau)$ for the evaluation of both predictors.

In Section 3.4.1, Table 3.1 shows that performance metrics p_0 and p are largest for input frequency of $\omega = 1.5$ rad/s and hence the closed-loop system is most sensitive to this frequency with presence of delays. Thus, a sinusoidal voltage input, $E = 12 \sin(1.5t)$ volts is given as the input to the system in the ideal, delayed and predictor cases and the simulation is run for 30 s as in Section 3.4.1. The performance metrics p_0 and p defined as above for each delay value and different λ are summarized in Table 3.2. When the delay increases, p_0 becomes larger and there exist a larger difference between the output trajectories of delayed and ideal cases. As shown in Figure 3.14,

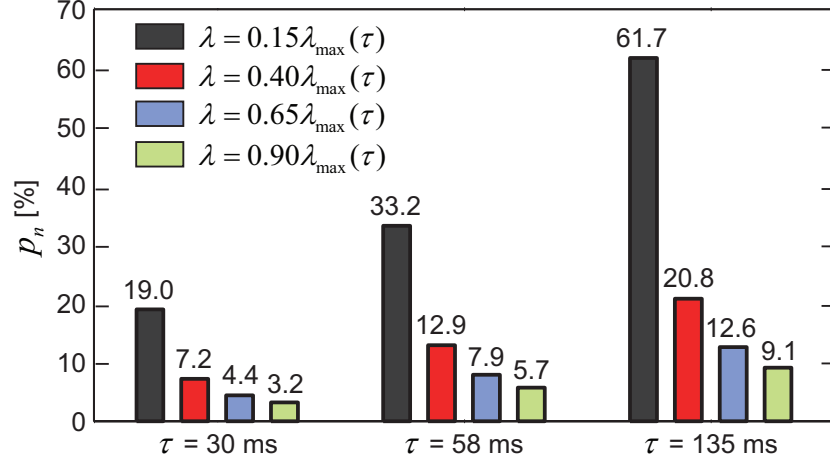


Figure 3.15: Normalized performance metrics p_n with different constant delays. Smaller p_n indicates better performance and higher fidelity in networked integration. For all delays, better performance is achieved with the predictors compared to the delayed case.

a constant delay of 135 ms significantly distorts the system output, with $p_0 = 26.31$ rad, while p_0 is only 7.61 rad with a delay of 58 ms.

The normalized performance metrics p_n are shown in Figure 3.15. All $p_n < 1$, indicating that predictors with different λ within the stability region improve the fidelity of the networked integration compared to the delayed case for all delays. With a selection of large λ , the output difference can be reduced to less than 10% of the difference caused by the delays. Figure 3.16 compares the output trajectories among all three cases when the one-way delay is 135 ms. With $\lambda = 0.90\lambda_{\max}(\tau)$, the output of the predictor case is very close to that of the ideal case and the predictors are very effective in terms of reducing the impact of delay.

3.4.2.2 Varying Delays

Similar simulations are run for varying delays. Measurements from three of the networks (AA-BE, AA-PA, AA-SH) in Section 3.2.2 are used as simulated delay models. Zero-order hold is applied to the measurements when there is a lack of data points due to packet dropouts through the network. Assuming that delays are the

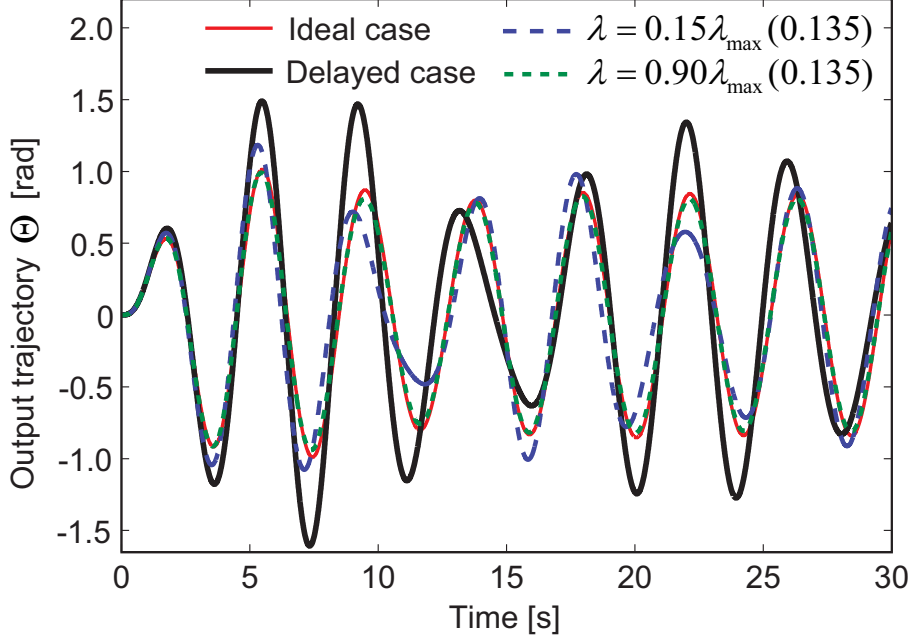


Figure 3.16: Output trajectory comparison among ideal case, delayed case with delays of 135 ms, and predictor case. Predictor cases with all λ lead to better performance compared to delayed case.

Table 3.3: Means and standard deviations of the round-trip delays for the three networks data

Network	AA - BE	AA - PA	AA - SH
Mean μ (ms)	30.2	58.1	135.6
Standard deviation σ (ms)	14.0	9.3	7.1

same in both directions, the mean and standard deviation of the one-way delay of the three networks are listed in Table 3.3.

The mean values in Table 3.3 are close to the three constant delays tested before, respectively. Recalling that the maximum stability range of the varying delay is smaller than the constant delay with the same mean value ($\frac{3}{2\tau_{\text{avg}}}$ and $\frac{\pi}{2\tau_{\text{avg}}}$, respectively), slightly larger scales of $\lambda_{\text{max}}(\tau_{\text{avg}})$ are chosen in the varying delay cases to make the λ values the same as in the constant delay cases. The performance metrics p_0 and p for each delay value are shown in Table 3.4 and the normalized performance metrics p_n are shown in Figure 3.17. Note that with varying delays, predictors still improve the fidelity of the networked integration when λ is chosen within the stability

Table 3.4: Performance metrics with different varying delays

Metrics \ Delay		One-way Varying Delay Mean τ_{avg}		
		30.2 ms	58.1 ms	135.6 ms
p_0 (rad)		3.57	7.64	25.91
p (rad)	$\lambda = 0.16\lambda_{\text{max}}(\tau_{\text{avg}})$	0.68	2.53	15.97
	$\lambda = 0.42\lambda_{\text{max}}(\tau_{\text{avg}})$	0.26	0.98	5.40
	$\lambda = 0.68\lambda_{\text{max}}(\tau_{\text{avg}})$	0.16	0.60	3.29
	$\lambda = 0.94\lambda_{\text{max}}(\tau_{\text{avg}})$	0.11	0.44	2.37

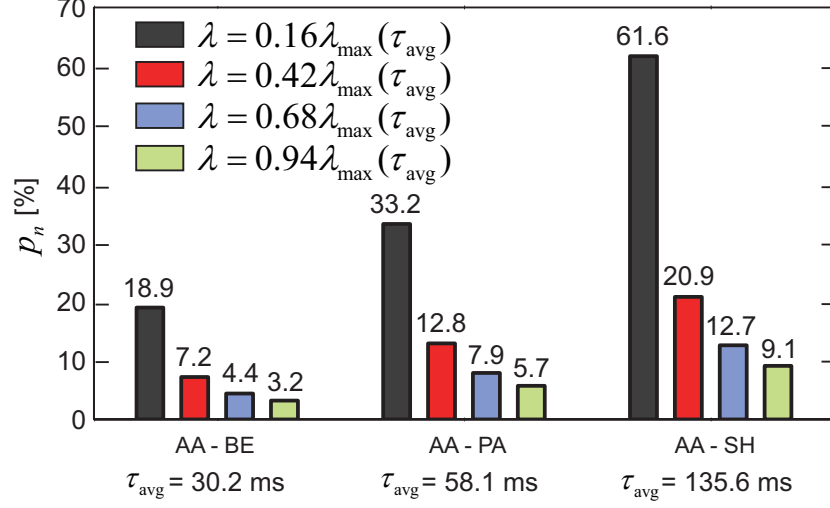


Figure 3.17: Normalized performance metrics p_n with different varying delays. Smaller p_n indicates better performance and higher fidelity in networked integration. For all delays, predictors with different λ improve the performance compared to the delayed case.

range. The levels of improvement are almost the same as in the constant delay cases because of two reasons. One reason is that both subsystems behave like low-pass filters, and the impact of the fast variation of delays on systems with low operating frequencies is not noticeable. Metrics p_0 without prediction tested under both constant and varying delays are very close. The second reason is the predictor dynamics, which is discussed in detail in the next section.

3.4.2.3 Discussion of Prediction Performance with Varying Delays

The prediction accuracy of Subsystem 2 predictor with varying delays is studied based on the simulation results using AA-SH network data as an example. Note

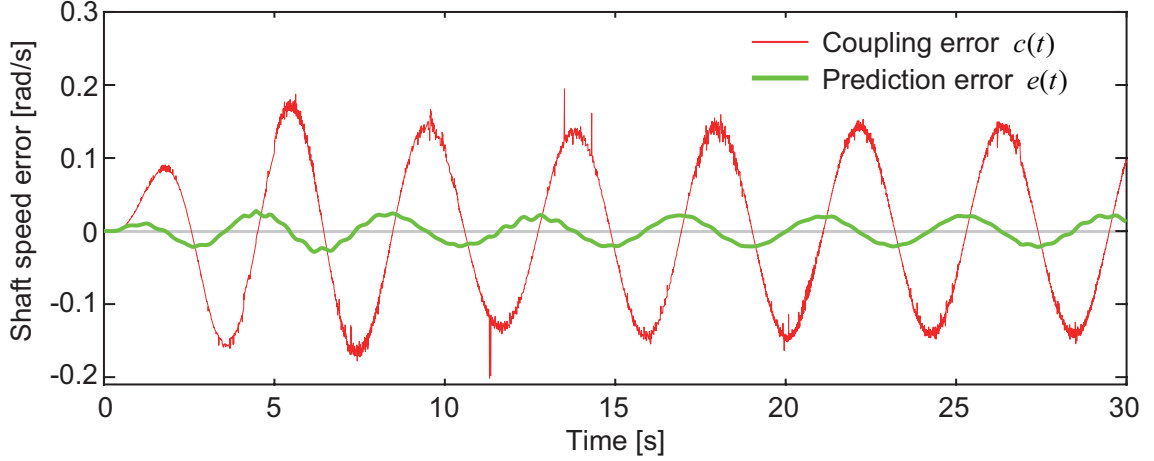


Figure 3.18: There exists fast varying jitters in the coupling error of shaft speed signal $y_2(t)$. Jitters are attenuated by Subsystem 2 predictor and the prediction error is smooth, resulting in a smooth predictor output without jitters.

that based on Table 3.3, the ratio of standard deviation of delays to the mean is around 0.05, which is very small and means that most of the delay values at each time instance is close to the mean delay value and only a few significant spikes are captured in the data.

Figure 3.18 shows the coupling error $c(t)$ between the shaft speed signal without and with delays $\tau(t)$. Due to variation of delays, packets enclosing unchanged information during transmission are received at different rates and may not be in the same order as when packets are sent, resulting in jittered predictor inputs including delayed signal $y(t - \tau(t))$ and its derivative $\dot{y}(t - \tau(t))$. Thus, $c(t)$ is not smooth and has significant spikes such as the ones at 11.3 s and 14.3 s. However, in Figure 3.18, the fast varying jitters are attenuated by Subsystem 2 predictor and the prediction error $e(t)$ is smooth. Therefore, a smooth predictor output is expected.

Figure 3.19 shows the comparison of different shaft speed signals among original $y_2(t)$, delayed $y_2(t - \tau(t))$, and Subsystem 2 predictor output $\hat{y}(t)$. Observed from the zoomed-in view of Figure 3.19, the predictor output $\hat{y}(t)$ is indeed smooth and is not affected much by the jitters in the inputs. One potential explanation is the novel predictor dynamics developed. Recall from the general predictor dynamics that

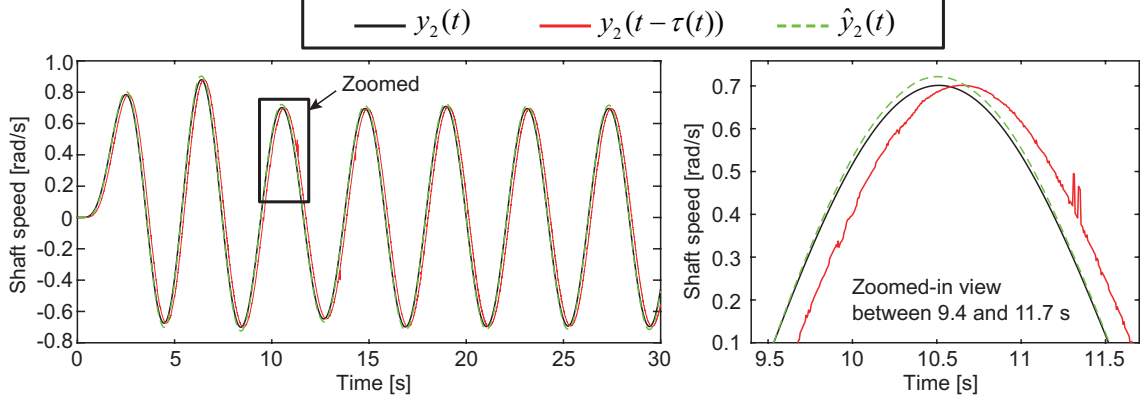


Figure 3.19: Comparison of shaft speed signals among undelayed $y_2(t)$, delayed $y_2(t - \tau(t))$, and Subsystem 2 predictor output $\hat{y}_2(t)$. Predictor has the potential of alleviating the jitters in the delayed signal and generating a smooth prediction $\hat{y}(t)$.

handles the inputs with varying delays $\tau(t)$ in (3.5). The jitters in $y(t - \tau(t))$ have an effect on determining the predictor state derivative $\dot{y}_p(t)$ instead of on $y_p(t)$ directly. Due to the integration action on $\dot{y}_p(t)$, $\hat{y}(t)$ is not very sensitive to the jitters within the predictor inputs $y(t - \tau(t))$.

In addition, it can be observed that the magnitude of $e(t)$ is much smaller than that of $c(t)$ in Figure 3.18 and $\hat{y}(t)$ is very close to the original $y(t)$ without delays Figure 3.19, both indicating good prediction accuracy using Subsystem 2 predictor to compensate effect of varying delays on the shaft speed signal. Similar results and conclusion are found in Subsystem 1 predictor, as well.

Thus, the predictor developed in this work has the potential of alleviating the jitters in the signal with varying delays and generating a smooth and accurate prediction.

3.4.3 Performance with Hardware Networks

In the final part of this case study, the example system is tested using a real network. With respect to Figure 3.9, the Subsystem 1 predictor and Subsystem 2 are run on one computer as the Server site, which is located in Alameda, California. Another computer acts as the Client site, running Subsystem 2 predictor as well as

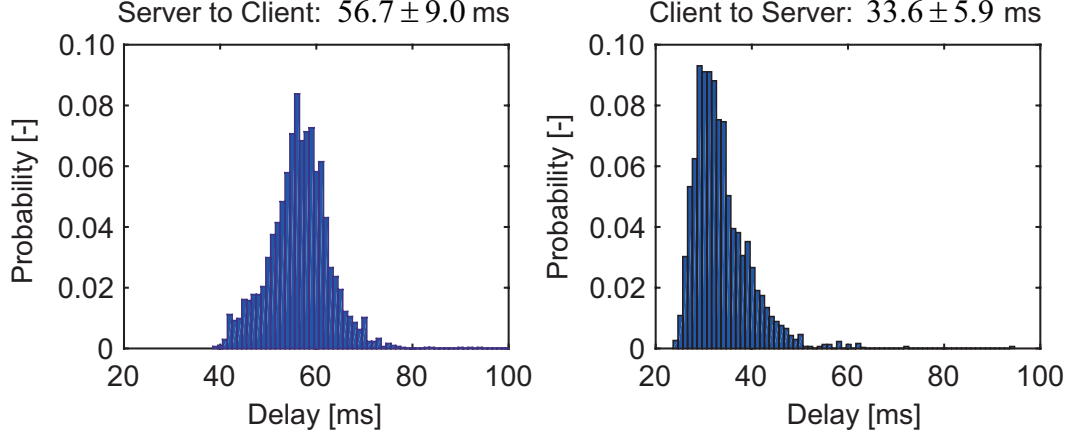


Figure 3.20: The histogram of one-way delay measurements (a) from Server to Client (b) from Client to Server. Round trip delay is 90.3 ± 9.3 ms.

Subsystem 1, and is located in Ann Arbor, Michigan. Both sites send communication packets through the Internet via wireless connections. This connection is referred as the MI-CA network in the following discussion. Packets are sent using the User Datagram Protocol (UDP) at the rate of 1 ms and the remote site responds whenever packets are received. In addition to $y_i(t)$ ($i \in \{1, 2\}$) and their derivatives $\dot{y}_i(t)$, the send time stamps of the local site are included in the packets as well. At the beginning of the simulation, the Network Time Protocol (NTP) [107] is used to synchronize the clocks of the two computers to the same NTP time server, so that the one-way packet delay can be calculated based on the difference between the send time and the receive time. Thus, this simulation with the real network includes the effects of not only the varying delays, but also distributed simulation, communication frequency, packet drops, delay measurement errors and clock synchronization errors.

The one-way delay distributions from Server to Client and from Client to Server in a 30 s period of time are shown in Figure 3.20 and the round trip delay is 90.3 ± 9.3 ms, with a drop rate of 14.6%. Simulating Server site and Client site simultaneously for 30 s, the performance metric p_0 for the delayed case without predictors is 5.04 rad. Including the predictors with λ to be 15%, 40%, 65% and 90% of $\lambda_{\max}(\tau_{\text{avg}}(t))$

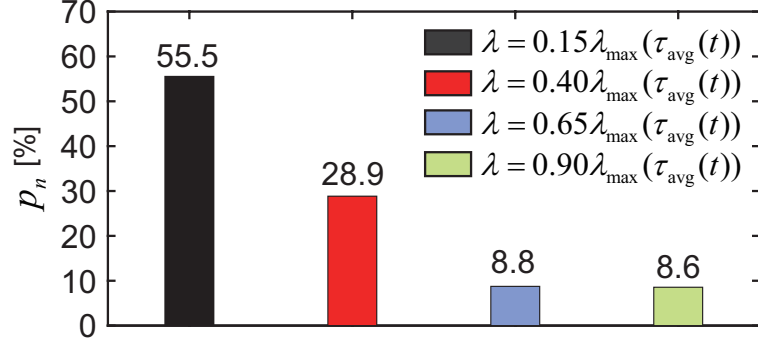


Figure 3.21: p_n with a real MI-CA network. Smaller p_n indicates better performance and higher fidelity in networked integration. Predictors with various λ improve the performance compared to the delayed case.

based on the 5 s moving average delays $\tau_{\text{avg}}(t)$, the performance metric p is reduced to 2.80, 1.45, 0.44, 0.43 rad, respectively. p_n are shown in Figure 3.21. All $p_n < 1$, indicating that predictors with various λ all improve performance and system fidelity over the real network with robustness against the aforementioned effects, such as the packet drops and delay measurement errors.

3.5 Closed-Loop System Stability with Predictors

Although predictor stability with constant and varying delays have been established, it remains unknown how the stability of the closed-loop system is affected by the predictors.

3.5.1 “Mixed” Passivity and Small Gain Method

To study the closed-loop system stability, a “mixed” passivity and small gain method [108] is leveraged in this paper. The method blends the passivity theory and small gain theory together so that a larger range of subsystems capturing the passivity property or small gain property or both, can be studied. The method is summarized as follows and is currently limited to study the stability of a linear time invariant (LTI) closed-loop system with constant delays.

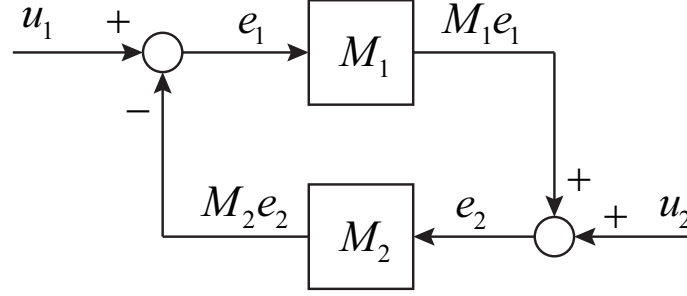


Figure 3.22: Interconnection of two nonlinear subsystems M_1, M_2

A pair of causal, bounded, linear (and not necessarily time-invariant) operators ($\Gamma : \mathcal{L}_2[0, \infty) \rightarrow \mathcal{L}_2[0, \infty)$ and $B : \mathcal{L}_2[0, \infty) \rightarrow \mathcal{L}_2[0, \infty)$) are introduced as weights to set up the strict “mixed” small gain and passivity property that is defined as [108]:

Definition III.1. If $\Gamma^* \Gamma + B^* B = \mathbf{I}$, system $M_i : \mathcal{L}_2 \rightarrow \mathcal{L}_2$ has a strict “mixed” small gain and passivity property if there exist constants $0 \leq \epsilon_i < 1, k_i > 0, l_i > 0, \eta_i \geq 0$ such that

$$\begin{aligned} -k_i \|B(M_i e_i)_t\|^2 + 2\langle B(M_i e_i)_t, B(e_i)_t \rangle - l_i \|B(e_i)_t\|^2 \\ - \|\Gamma(M_i e_i)_t\|^2 + \epsilon_i \|\Gamma(e_i)_t\|^2 + \eta_i \geq 0 \end{aligned} \quad (3.32)$$

for all input $e_i \in \mathcal{L}_2$ and all $t \geq 0$, where $M_i e_i$ is the output and the subscript t represents the signal at instant t .

Note that if $\Gamma = \mathbf{0}$, (3.32) indicates M_i with strict input and output passivity and if $B = \mathbf{0}$, (3.32) indicates M_i with gain < 1 .

Consider an interconnection of two general nonlinear systems M_1, M_2 in Figure 3.22. Theorem 5 in [108] states that if (3.32) holds for $i = \{1, 2\}$ and there exist constants satisfying:

$$\begin{aligned} 0 \leq \epsilon_1 \leq 1 \quad 0 \leq \epsilon_2 \leq 1 \\ l_1 + k_2 \geq 0 \quad l_2 + k_1 > 0 \quad k_2 > 0 \quad l_2 > 0 \end{aligned} \quad (3.33)$$

then $u_1, u_2 \in \mathcal{L}_2[0, \infty) \Rightarrow e_1, e_2, M_1 e_1, M_2 e_2 \in \mathcal{L}_2[0, \infty)$. In other words, when M_1

and M_2 satisfy the “mixed” small gain and passivity property in Definition III.1 (M_2 satisfies the strict version of it), the closed-loop system (interconnection of M_1 and M_2) is input-output stable.

Note that the above theorem applies to nonlinear, multi-input multi-output and time varying systems in general. It can be tested based on the measured data of inputs e_i and outputs $M_i e_i$ [109], without necessarily requiring the information of systems M_1 and M_2 . However, the limitations of the experimental approach are obvious: no matter how large the amount of data is, they cannot cover the whole input set $\{(u_1, u_2) : u_1, u_2 \in \mathcal{L}_2[0, \infty)\}$ and infinite time horizon in practice.

In special conditions such as LTI systems, the “mixed” property in Definition III.1 is equivalent to the system being passive at low frequencies and having small gain at high frequencies, when Γ is time invariant with $|\Gamma(j\omega)|$ close to 0 at low frequencies and close to 1 at high frequencies [108, 110]. Thus, Theorem 5 in [108] is leveraged as follows:

Theorem III.2. *If each of the two systems M_1, M_2 at all frequencies is input and output strictly passive or has gain less than one or satisfies both, then the interconnection of M_1 and M_2 is stable.*

In other words, with the frequency dependent form of passivity and small gain property for LTI systems stated as:

- Passivity Property:

$$\operatorname{Re}(M_i(j\omega)) > 0 \text{ for } |\omega| \in [0, \omega_a) \quad (3.34)$$

- Small Gain Property:

$$|M_1(j\omega)| \cdot |M_2(j\omega)| < 1 \text{ for } |\omega| \in (\omega_b, \infty) \quad (3.35)$$

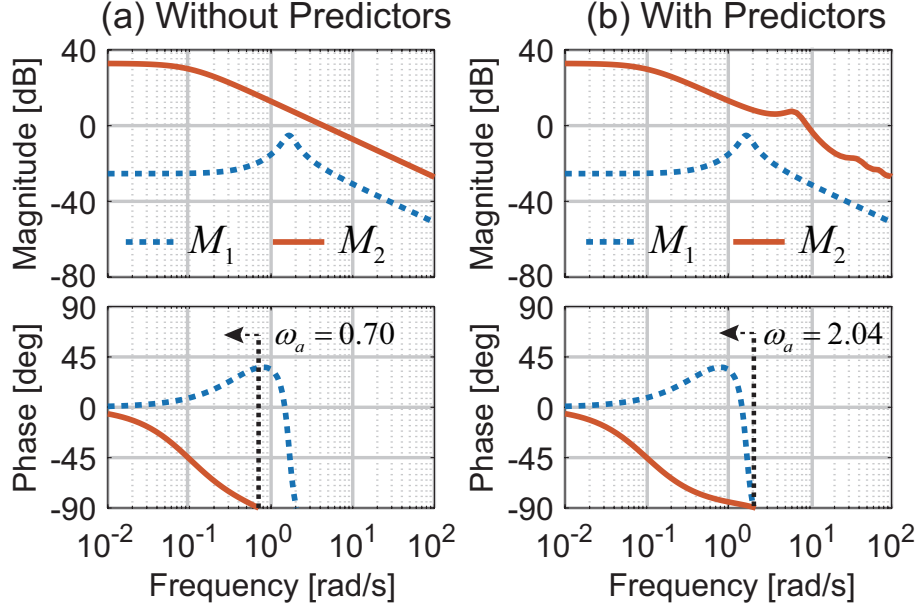


Figure 3.23: Bode plots of M_1 and M_2 (a) without predictors (b) with subsystem 2 predictor $\lambda = 0.60\lambda_{\max}(0.2)$. Predictor extends the frequency bandwidth within which the system is passive.

if $\omega_a > \omega_b > 0$, the closed-loop system in Figure 3.22 is stable.

In this dissertation, Theorem III.2 with respect to LTI systems is leveraged to establish the stability of the example LTI motor-shaft-torque networked closed-loop system with predictors and constant delays shown in Figure 3.9 to start with and could be applied to teleoperated vehicle system of interest in the future. Without collecting massive data of the inputs and outputs of each subsystem, the closed-loop stability is analyzed by relying on the subsystem model information. Theorem III.2 provides some insight into how delays destabilize the system and how predictors can help with re-establishing the closed-loop stability. The following simulation results demonstrate how the predictors can stabilize a networked system that is otherwise unstable due to delays.

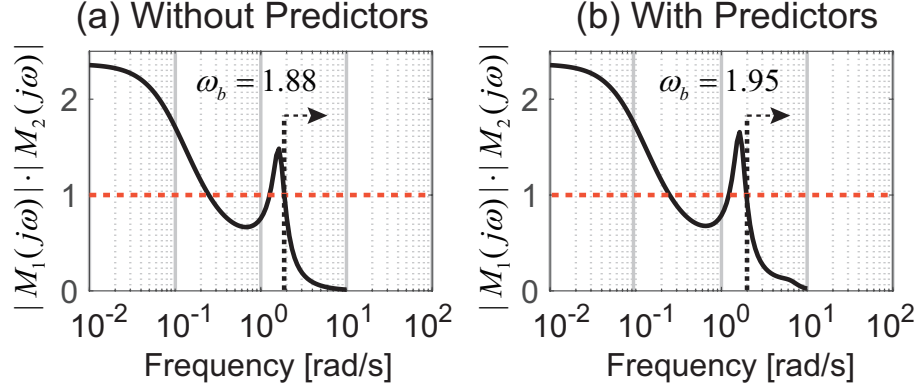


Figure 3.24: The product of gain $|M_1(j\omega)| \cdot |M_2(j\omega)|$ (a) without predictors (b) with subsystem 2 predictor $\lambda = 0.60\lambda_{\max}(0.2)$. Predictor slightly amplify the system gain.

3.5.2 Simulation Results

Note that the example networked system without predictors can tolerate one-way delays of up to 175 ms to remain input-output stable. A constant delay beyond the stability limit of the networked system is considered to demonstrate the ability of the predictor framework to stabilize the system. Consider the one-way delay of 200 ms and Theorem III.2. For the example system in Figure 3.9, combining Subsystem 1, Delay 1 and Subsystem 1 predictor as system M_1 and the remaining components related to Subsystem 2 as system M_2 , transfer functions $M_1(s)$ and $M_2(s)$ are defined as:

$$\begin{aligned}
 M_1(s) &= G_{\text{Sys1}}(s)G_{\text{Delay1}}(s)G_{\text{Pred1}}(s) \\
 &= \frac{0.28s + 0.1484}{s^2 + 0.53s + 2.75} e^{-0.2s} \frac{s + \lambda_1}{s + \lambda_1 e^{-0.2s}} \\
 M_2(s) &= G_{\text{Sys2}}(s)G_{\text{Delay2}}(s)G_{\text{Pred2}}(s) \\
 &= \frac{4.4}{s + 0.1} e^{-0.2s} \frac{s + \lambda_2}{s + \lambda_2 e^{-0.2s}}
 \end{aligned} \tag{3.36}$$

and the closed-loop system aligns with the form in Figure 3.22. In the delayed case when predictors are not involved, $\lambda_1 = \lambda_2 = 0$. Bode plots of $M_1(s)$ and $M_2(s)$ without predictors are shown in Figure 3.23(a). M_1 is passive (i.e., phase is within $(-90^\circ, 90^\circ)$) when $|\omega| \in [0, 2.04)$ and M_2 is passive when $|\omega| \in [0, 0.70)$. The common

frequency limit in Theorem 1 when M_1 and M_2 are both passive is $\omega_a = 0.70$.

The small gain property is checked by plotting the product of each system gain, $G_{CL}(\omega) = |M_1(j\omega)| \cdot |M_2(j\omega)|$. In Figure 3.24(a), $G_{CL}(\omega) < 1$ when $\omega > 1.88$ rad/s. Thus, without predictors, the frequency limit in Theorem 1 when interconnection of M_1 and M_2 has gain less than 1 is $\omega_b = 1.88$.

Since $\omega_a < \omega_b$, the “mixed” small gain and passivity property does not hold for all the frequencies, and the closed-loop system is in fact unstable as seen in Figure 3.25. To stabilize the system, predictors are included to alter ω_a and ω_b . Note that from Figure 3.23(a), the frequency range when M_2 is passive is very limited compared to that of M_1 . Therefore, only Subsystem 2 Predictor is included to demonstrate the predictor’s capability of stabilizing an unstable closed-loop system in this example. With $\lambda_2 = 0.60\lambda_{\max}(0.2)$, the updated Bode plots of M_1 and M_2 are shown in Figure 3.23(b). Subsystem 2 Predictor in M_2 significantly extends the frequency bandwidth for passive M_2 to that of M_1 and ω_a is increased to 2.04. Accordingly, in Figure 3.24(b), the predictor slightly amplifies the system gain and ω_b is now 1.95. Note that in this case, $\omega_a > \omega_b$. Based on Theorem 1, the closed-loop stability is achieved.

Closed-loop simulations are performed next to validate the theorem. As Figure 3.25 shows, the closed-loop system output (marked in thick and solid black) is unstable without the predictor. With the predictor involved, when $\lambda_2 = 0.60\lambda_{\max}(0.2)$, closed-loop system regains stability due to Subsystem 2 Predictor. Furthermore, system output in this case is very close to the ideal case without delay. $p_n = 0.18$ is significantly below 1, meaning the networked system fidelity is greatly improved. The predictor’s capability of re-establishing the closed-loop system stability is thus demonstrated.

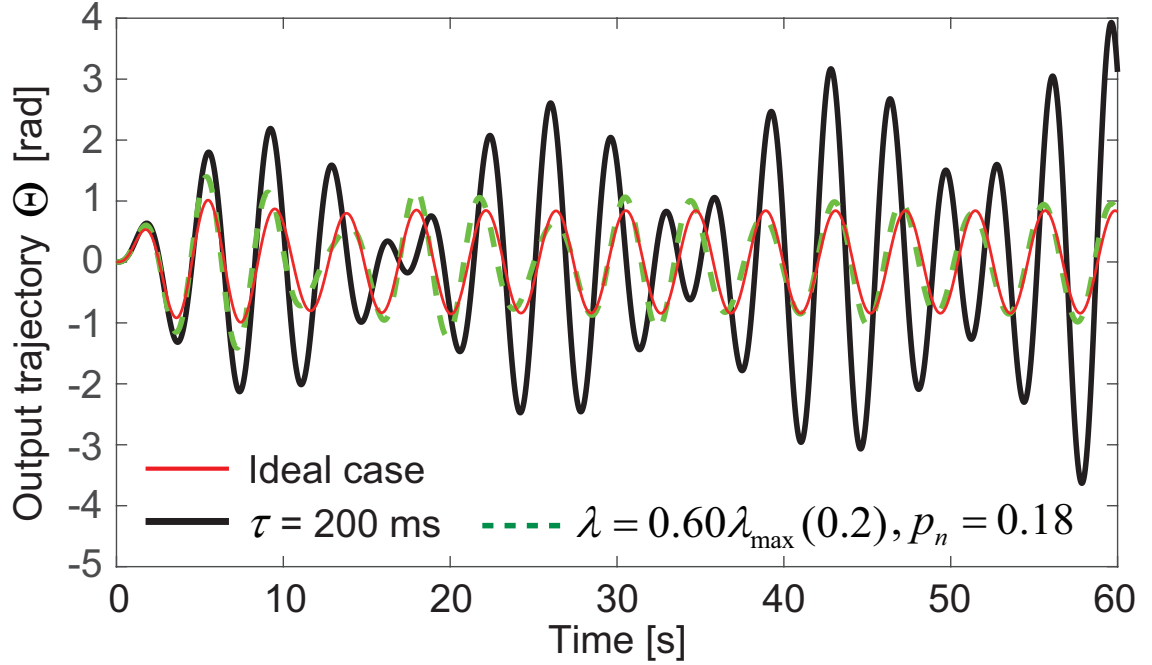


Figure 3.25: When the delays make the closed-loop system unstable, predictors manage to stabilize the system and improve the fidelity as well.

3.6 Saturation and Resetting Scheme

Observed from an example comparison between predictor output and original signal $y(t)$ without delays in Figure 3.19, there always exists overshoots when $y(t)$ changes direction. Since only the delayed inputs $\dot{y}(t-\tau(t))$ and $y(t-\tau(t))$ are available to the predictor, it cannot recognize the direction changes of $y(t)$ in time and the peak of $y_p(t)$ lags the peak of $y(t)$, thereby generating the overshoot. Overshoots with large magnitudes could contribute to performance loss in the applications where transient response is critical. In teleoperated vehicle system, human operators rely on the predicted vehicle heading, speed and location to generate consequent commands to control the vehicle, and large overshoots may cause significant errors in the graphical display of the predicted vehicle heading, which in turn can cause the operators steer more aggressively and exacerbate the overshoots afterwards.

Thus, the concern of overshoot inspires the development of a saturation and resetting scheme in addition to the existing predictor dynamics to improve predictor

transient performance. To better illustrate the scheme, an example of predicting a signal of vehicle heading is provided.

The heading signal to be predicted is from a human-in-the-loop experiment [111] under the teleoperated vehicle system setup as in Figure 2.1, where a human operator controlled the steering and speed of a simulated vehicle in a track-following scenario, with constant control delays of 0.3 s and constant sensor delays of 0.6 s. Data was recorded at the sample rate of 100 Hz for 60 s and includes control commands of throttle, brake and steering, as well as vehicle states of heading, location and speed. The undelayed heading $y(t)$ and delayed heading with delays of 0.6 s, i.e. $y(t - 0.6)$, within a time window between 28 s and 38 s are shown in Figure 3.26. Predictor with a intuitively selected $\lambda = 0.40\lambda_{\max}(0.6)$ is implemented and there exists overshoots with large magnitude and long duration when $y(t)$ changes direction. The saturation and resetting scheme designed for the predictor in this work includes two stages, namely, output saturation and state resetting, which are illustrated in detail separately below.

The first stage is predictor output saturation, where the condition of detecting the overshoot sooner is studied and then the predictor output is saturated to reduce the magnitude of overshoot.

Let $\hat{y}_{\text{sat}}(t)$ be defined as

$$\hat{y}_{\text{sat}}(t) := \frac{\dot{y}(t - \tau(t))}{\lambda} + y(t - \tau(t)) \quad (3.37)$$

Comparing the current predictor output $\hat{y}(t)$ to $\hat{y}_{\text{sat}}(t)$, the peak of $y(t)$ can be detected sooner. For example, in Figure 3.26, $\hat{y}(t) = \hat{y}_{\text{sat}}(t)$ at 29.6 s, which detects the peak 0.4 s sooner than the original predictor, where the peak is detected only when $y_p(t - \tau(t)) = \hat{y}_{\text{sat}}(t)$ at 30 s, which causes a significant overshoot as the gray dashed line in Figure 3.26 illustrates. Therefore, at 29.6 s, predictor output $\hat{y}(t)$ can be saturated to be no greater than $\hat{y}_{\text{sat}}(t)$ to reduce the overshoot. Similarly, $\hat{y}_{\text{sat}}(t)$

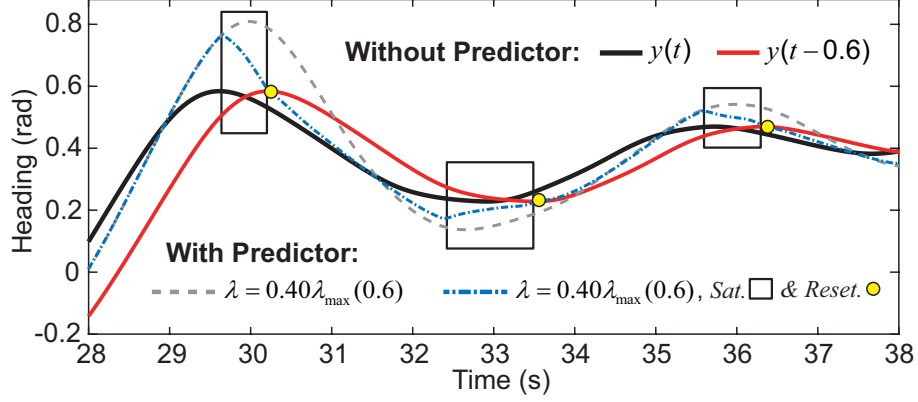


Figure 3.26: Comparison of different heading signals. When applying a saturation and resetting scheme with the predictor, the overshoot in the transient is reduced and thus leads to better predictor transient performance.

can be used as a lower limit of $\hat{y}(t)$ when $\dot{y}(t - \tau(t)) < 0$. Such saturation is also applicable when there is no overshoot, because if $y(t)$ is monotonically changing, $\hat{y}_{\text{sat}}(t)$ expands faster than $\hat{y}(t)$ and the saturation would not be triggered. Therefore, the predictor dynamics and output function of (3.5) are modified as follows to include the described output saturation:

$$\begin{aligned}\dot{y}_p(t) &= \dot{y}(t - \tau(t)) + \lambda[y(t - \tau(t)) - \hat{y}(t - \tau(t))] \\ \hat{y}(t) &= f_{\text{sat}}(y_p(t), \dot{y}(t - \tau(t)))\end{aligned}\tag{3.38}$$

where $f_{\text{sat}}(y_p(t), \dot{y}(t - \tau(t)))$ deals with saturation according to

$$f_{\text{sat}}(y_p(t), \dot{y}(t - \tau(t))) = \begin{cases} \min(y_p(t), \hat{y}_{\text{sat}}(t)) & \text{if } \dot{y}(t - \tau(t)) \geq 0 \\ \max(y_p(t), \hat{y}_{\text{sat}}(t)) & \text{if } \dot{y}(t - \tau(t)) < 0 \end{cases}\tag{3.39}$$

and $\hat{y}_{\text{sat}}(t)$ is updated using (3.37) at each time instance.

The second stage is predictor state resetting. Note that in the modified form (3.38), output $\hat{y}(t)$ and state $y_p(t)$ could be different from each other. Predictor state is reset to the delayed signal input $y(t - \tau(t))$ after the overshoot, so that $\hat{y}(t)$ and $y_p(t)$ match again, and (3.38) becomes the same as the original form (3.5), for which

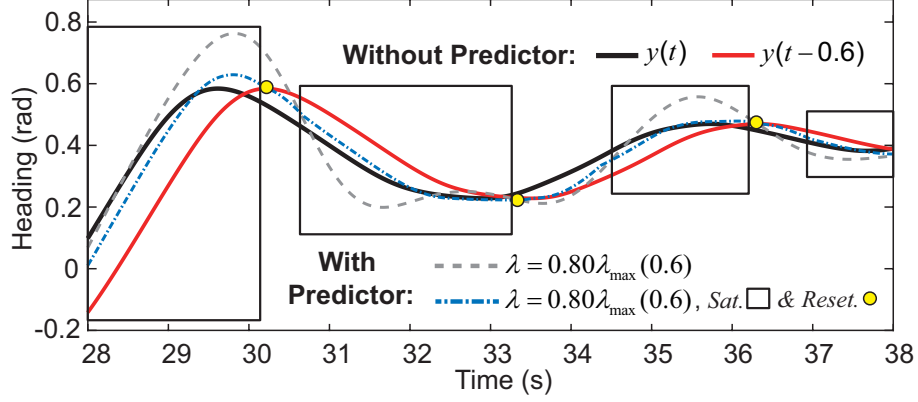


Figure 3.27: There exists oscillation in the predictor output when compensating large delays of 0.6 s with large λ . Applying the saturation and resetting scheme helps reduce the oscillation introduced by the predictor.

stability and performance characteristics are well established. In Figure 3.26, between 29.6 s and 30.2 s, $\dot{y}(t - \tau(t)) > 0$ and $\hat{y}(t)$ is saturated, i.e. $y_p(t) \geq \hat{y}_{\text{sat}}(t)$. At 30.2 s, $\dot{y}(t - \tau(t))$ becomes 0 and starts to change sign, indicating a direction change in the delayed input, and the overshoot is considered to end here. $\hat{y}_{\text{sat}}(t) = y(t - \tau(t))$ based on (3.37) and the predictor state $y_p(t)$ is reset back to the delayed input $y(t - \tau(t))$, as well. Thus, output $\hat{y}(t)$ and state $y_p(t)$ match with each other and a new prediction is started afterwards. For $\dot{y}(t - \tau(t)) < 0$, similar resetting occurs after the overshoot between 32.3 s and 33.6 s.

The state reset condition is generalized as follows: predictor state $y_p(t)$ is reset to be the same as delayed input $y(t - \tau(t))$ when

$$\begin{cases} \dot{y}(t - \tau(t)) \text{ changes from } (+) \text{ to } (-) \text{ and } y_p(t) \geq \hat{y}_{\text{sat}}(t) \\ \dot{y}(t - \tau(t)) \text{ changes from } (-) \text{ to } (+) \text{ and } y_p(t) < \hat{y}_{\text{sat}}(t) \end{cases} \quad (3.40)$$

Both output saturation and state resetting are used together. In Figure 3.26, where saturations and resets take action are marked using black rectangles and yellow dots, respectively. Applying the saturation and resetting scheme with $\lambda = 0.40\lambda_{\text{max}}(0.6)$, the predictor output heading within the time window between 28 s

and 38 s is closer to $y(t)$ than that without applying this scheme.

Additionally, the saturation and resetting scheme has the benefit of smoothing the prediction. A larger $\lambda = 0.80\lambda_{\max}(0.6)$ is selected and the predictor output within the same window of interest between 28 s and 38 s is shown in Figure 3.27. Unlike the previous prediction with small $\lambda = 0.40\lambda_{\max}(0.6)$, there exists significant amount of oscillation due to the large gain λ in the dynamics. The same saturation and resetting scheme is applied on this predictor, and black rectangular windows and yellow dots represent when the saturation stages and resetting stages are happening, respectively. Predictor output saturation to $\hat{y}_{\text{sat}}(t)$ occurs for majority of the time. Since $\hat{y}_{\text{sat}}(t)$ is updated based on (3.37), $\hat{y}_{\text{sat}}(t)$ is very close to delayed input signal $y(t - \tau(t))$ with large λ . Thus, the final predictor output may not completely match with undelayed signal, but at least is smooth in transient and does not have much additional oscillation introduced by the predictor.

In summary, applying the saturation and resetting scheme with the original predictor helps reduce the overshoot and oscillation in the predictor output, thereby improving transient performance. However, one drawback is that the final output with the scheme is more sensitive to the potential noise or jitter in the delayed inputs, because the conditions that triggers output saturation and state resetting highly depend on the sign of $\dot{y}(t - \tau(t))$.

3.7 Predictor Design Procedure

Predictors have been proven helpful in compensating constant and varying delays by generating signal prediction and potentially improving closed-loop system performance based on results of the case study in Section 3.4. In this section, a general predictor design procedure that systematically selects its only parameter λ to predict any signal originating from any system is provided to achieve a good performance metric and ensure predictor stability. Note that the procedure is developed for con-

stant delay case and could potentially be used as a reference for the varying delay case, when considering the moving average of varying delays. The design procedure has three major steps.

Step 1: Estimate τ , Range of ω_p , and ω_c

The first step is to determine or estimate the one-way delay τ to compensate, range of predictor bandwidth ω_p , and coupling error bandwidth ω_c that contains the dominant frequencies of $c(t)$.

The delay τ can be measured by calculating the difference between the send and receipt time stamps of the packets that are exchanged between the two systems connected via the communication network. To that end, methods such as the network time protocol [107] can be leveraged to synchronize the clocks on both sides.

The predictor bandwidth ω_p depends on values of τ and λ , i.e. $\lambda = 2\omega_p \sin(\tau\omega_p)$ in (3.25). With a knowledge of delay τ , the maximum ω_p is $\frac{0.959}{\tau}$ shown in (3.26) and larger λ leads to larger ω_p .

The coupling error bandwidth ω_c can be estimated based on analyzing the power spectral density (PSD) of coupling error $c(t)$. This requires some representative data of the signal to predict and calculation of $c(t)$ between undelayed and delayed signal with known delays. PSD is performed on $c(t)$, and ω_c is determined based on the frequency range where majority of the signal power is distributed. For example, ω_c can be considered as the frequency when the integrated power of the signal spectrum up to that frequency reaches 99% of the total signal power within the Nyquist frequency of $c(t)$.

Step 2: Identify a range of λ based on Stability and Steady State Performance

The second step is to identify a range of λ based on the relationship between ω_c

and ω_p , while considering the predictor stability and steady state performance.

If ω_c is small, it is possible to select a λ based on (3.25) and (3.26) such that $\omega_p \geq \omega_c$ and predictor is effective for all the dominant frequencies of $c(t)$.

For the case when $\omega_p < \omega_c$, two strategies are conceived. The first one is to delay the internal predictor state $y_p(t)$ in (3.5) by a less amount τ_p than the actual measurement of delay value, τ . With this strategy, although a lag of at least $\tau - \tau_p$ would still remain in the predictor output compared to the undelayed signal, smaller delay value leads to larger ω_p and there is higher chance to select a λ such that $\omega_p \geq \omega_c$. Therefore, predictor performance is effective in improving steady state performance while compensating partial delays in the signal.

The second strategy is to use the full amount of the actual delay τ in the predictor dynamics (3.5), but selecting a small λ to reduce significant amplification of $c(t)$ for $\omega > \omega_p$ based on plots of $M(\omega)$. Using such strategy, transient performance may be worse due to smaller λ , but the generated predictor output will match the undelayed signal without a constant lag when signal is close to reach the steady state.

Note that there is no apparent preference between these two strategies, as one may outperform the other when predicting a certain signal and could be visa versa for predicting another signals. The decision of which strategy to apply again relies on the relative magnitude between potential ω_p and ω_c .

Step 3: Select λ based on Transient Performance

In the final step, given a suitable range of λ that ensures predictor stability and improves steady state performance, λ is selected by further considering the transient performance of the predictor.

Larger λ usually results in a faster transient, but exhibits oscillations if λ is selected to be too large. Also, the overshoots when there exists direction changes in the signal are also larger with larger λ . To reduce the overshoot in the transient, a

saturation and resetting scheme presented in Section 3.6 is available, but may be sensitive to noise or jitter in the predictor input of delayed signal derivatives. Applying the saturation and resetting scheme is thus a design decision to make, as well.

Summary of Predictor Design Procedure

For each transmitted signal to predict, the procedure of designing the predictor is summarized as follows:

1. Estimate one-way delays τ , range of predictor bandwidth ω_p , and coupling error bandwidth ω_c .
2. Choose a range for λ based on the magnitude $M(\omega)$ in terms of predictor steady state performance, such that the predictor bandwidth $\omega_p > \omega_c$. If no λ within the predictor stability range (3.20) is found, apply either Strategy 1 or Strategy 2.
3. Determine λ according to the transient performance of the predictor. Apply the optional saturation and resetting scheme to reduce overshoots in the transients.

This predictor design procedure is used in the later sections to design predictors to predict the control commands and/or vehicle states in a teleoperated vehicle system. In Section 4.2, a detailed example of showing the design procedure and decision making in the process of selecting λ is illustrated to help understand the procedure.

CHAPTER IV

A Predictor Based Framework with Blended Prediction Architecture

In this chapter, a novel predictor based framework is developed for the teleoperated vehicle system to aim for improving vehicle mobility with robustness. The framework allows bilateral prediction on control commands and vehicle states and includes a blended prediction architecture to provide prediction of vehicle heading while emphasizing both the prediction accuracy and robustness. The framework with the blended prediction architecture is introduced first and the enclosed details are then illustrated. An open-loop test is performed in simulation to evaluate the vehicle heading prediction accuracy with blended architecture compared to using either the model-free predictors or model-based *feedforward branch* alone. The work in this chapter is based on publication [84].

4.1 Structure of a Novel Predictor Based Framework

In Chap II, a gap is concluded for the existing prediction based methods for delay compensation. In short, model-based prediction may sacrifice robustness to modeling errors, while the accuracy of model-free prediction depends on how representative either the assumptions about the particular way the vehicle moves, or knowledge

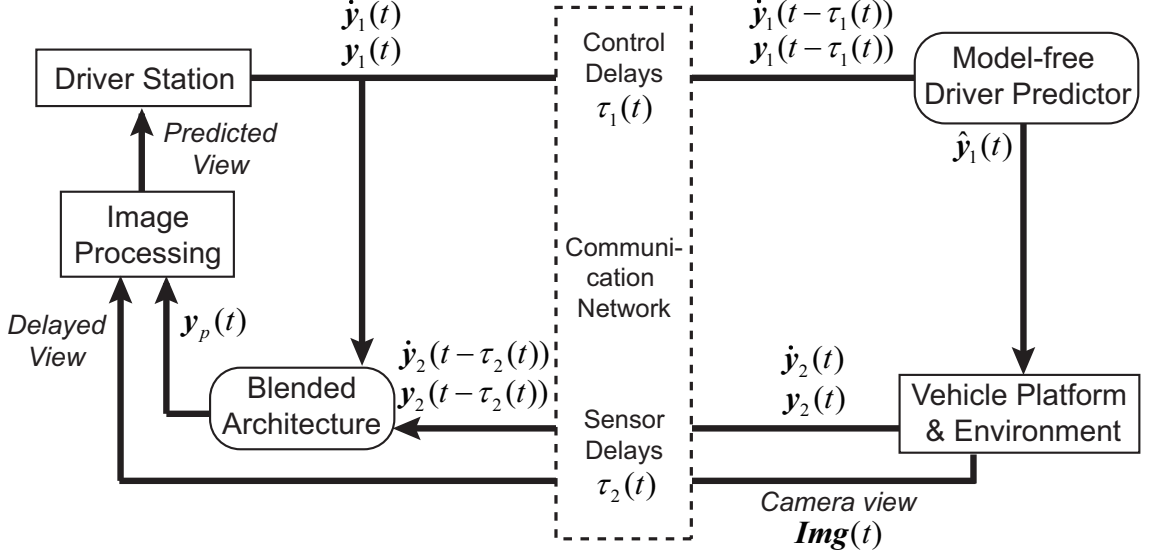


Figure 4.1: A novel prediction based framework to compensate delays.

about the noise characteristics that drive the existing predictive filters. To fill in the gap, this dissertation develops a novel predictor based framework with a blended prediction architecture for both the prediction accuracy and robustness to modeling errors.

The framework is shown in Figure 4.1. The transmitted signals are shown as vectors in bold fonts. Bilateral predictions on control commands $\mathbf{y}_1(t)$ and vehicle states $\mathbf{y}_2(t)$ are performed. Bilateral predictions are expected to be more beneficial in practice, since prediction in each way deals with less amount of delay ($\tau_1(t)$ or $\tau_2(t)$) than the round trip delay $\tau_{\text{RTT}}(t)$. One modification is needed compared to the generic teleoperated vehicle system setup in Figure 2.1: In addition to the original transmitted signals $\mathbf{y}_1(t)$ and $\mathbf{y}_2(t)$, their derivatives, $\dot{\mathbf{y}}_i(t)$ ($i \in \{1, 2\}$), as well as the send time stamp right before transmission are included in the communication packets. The reason to include them in packets is because the model-free predictor developed in this work requires the information of the signal derivatives with delays, i.e. $\dot{\mathbf{y}}_i(t - \tau_i(t))$, and one-way delay value $\tau_i(t)$ to perform prediction.

The framework with bilateral prediction is designed to leverage minimal informa-

tion about the existing system (i.e. human operator or vehicle dynamics) to improve performance of vehicle mobility with robustness. In the direction from Driver Station to Vehicle Platform, a model-free driver predictor is placed at the vehicle side to generate prediction of control commands $\hat{\mathbf{y}}_1(t)$. The details of the predictor is explained in Chapter III. In the direction from Vehicle Platform to Driver Station, the control command sequences $\mathbf{y}_1(t)$ and received vehicle states and states derivatives with sensor delays (i.e. $\mathbf{y}_2(t - \tau_2(t)), \dot{\mathbf{y}}_2(t - \tau_2(t))$) are available at the driver station side and a blended architecture is first proposed to generate prediction of vehicle states $\hat{\mathbf{y}}_2(t)$. Note that transmitted vehicle states required for teleoperated driving include vehicle position and heading to provide the information of vehicle movement in the environment, as well as vehicle longitudinal speed that is displayed to human operators and thus controlled by them using throttle and brake pedals. The delayed camera view is then altered to show the predicted states $\mathbf{y}_p(t)$ containing position, heading and speed in form of vision by leveraging existing image processing methods and displayed to the human operation for closed-loop teleoperated driving.

The blended architecture is shown in Figure 4.2. It is at the Driver Station and is composed of a *feedback branch*, a *feedforward branch* and a linear blending between the outputs of both branches for vehicle heading prediction. Compared to the structure of vehicle states prediction in the predictive display approach or human operation prediction or a mixture of both for bilaterally prediction, where either model-based or model-free prediction method is used, the main feature of the proposed blended architecture is to combine both a model-based *feedforward branch* and a model-free *feedback branch* for the sake of improving prediction accuracy with robustness.

Within the *feedback branch (FB)*, a model-free Vehicle Predictor based on the same dynamics and analysis as the Driver Predictor is employed, receiving the delayed vehicle states and state derivatives as inputs and generating predicted output vectors $\hat{\mathbf{y}}_2(t)$. $\hat{\mathbf{y}}_2(t)$ contains predicted 2D positions $X_p(t)$ and $Y_p(t)$, predicted lon-

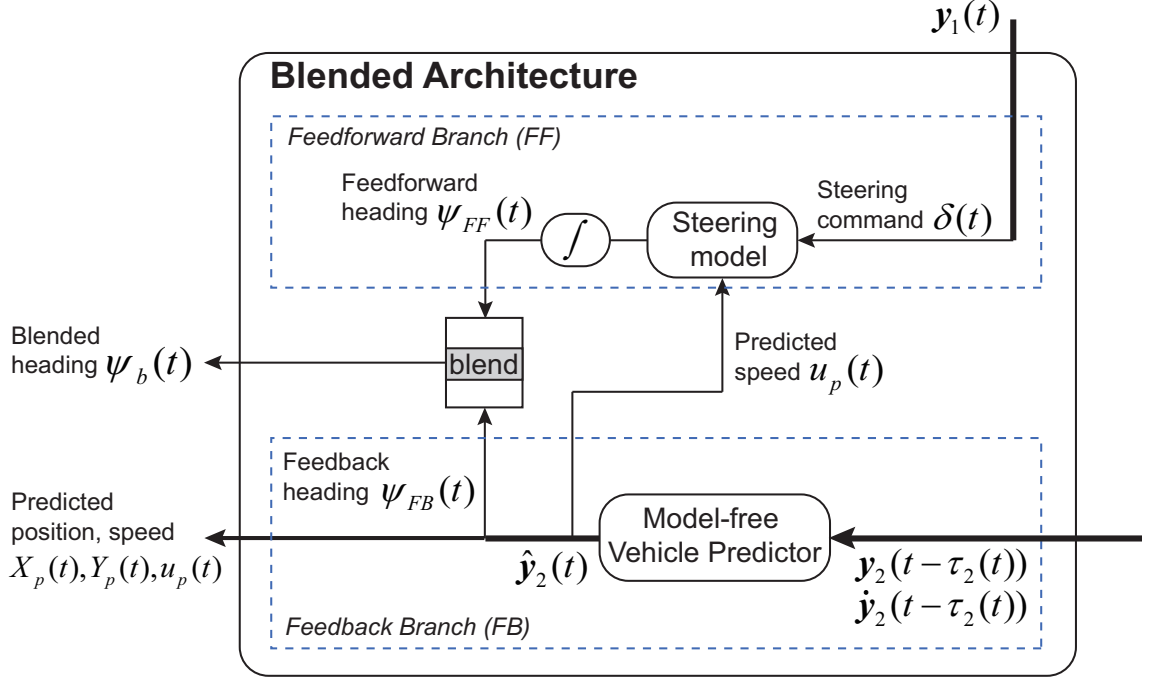


Figure 4.2: The blended architecture is composed of a *feedback branch*, a *feedforward branch* and a linear blending for vehicle heading predicting. Minimal vehicle information is relied on to generated prediction of vehicle states with robustness.

gitudinal speed $u_p(t)$ and feedback heading $\psi_{FB}(t)$. Since predictions are performed based on the vehicle states that are the feedback outputs from the remote vehicle platform, the branch is named as *feedback branch*. Note that this branch can generate prediction of all the vehicle states. When implemented with the model-free Driver Predictor in predicting control commands, the framework is completely model free in the sense that no knowledge about the governing equations of the vehicle platform or human operator is required to perform the predictions of all transmitted signals and there exists no robustness issue to modeling errors compared to model-based prediction. Besides, no assumptions about the vehicle trajectory or human motion and no knowledge or estimation of noise in the high order derivatives of the transmitted signals are required compared to existing model-free prediction methods. However, the limitation of the model-free predictor developed in this dissertation is that their prediction accuracy may not be comparable to the model-based methods when accurate

models are available, due to limited predictor bandwidth to predict high frequency components of the signals under large delays.

Thus, a model-based *feedforward branch* as well as the linear blending are developed to help further improve prediction accuracy of vehicle heading, by incorporating minimal information of vehicle steering dynamics while preserving as much robustness as possible. The reason of adding the blending for heading prediction only is that early experiments showed that human operators were affected more by the asynchrony between steering and monitoring the subsequent vehicle heading than by the asynchrony between controlling and monitoring vehicle longitudinal speed [18, 111].

Within the *feedforward branch* (*FF*), there exists a steering feedforward loop where feedforward heading $\psi_{FF}(t)$ is generated based on the human operators current steering command $\delta(t)$. A steering model that represents the lateral dynamics of the vehicle platform is needed to map the steering to heading; hence this model-based branch may suffer from low robustness to modeling errors.

Given that each branch has its own advantages and limitations, and recognizing the potential that the two branches can synergistically complement each other, their outputs of heading $\psi_{FF}(t)$ and $\psi_{FB}(t)$ are linearly combined as follows to create a blended architecture:

$$\psi_b(t) = (1 - \alpha(t))\psi_{FF}(t) + \alpha(t)\psi_{FB}(t) \quad (4.1)$$

where $\psi_b(t)$ is the resulting blended predicted heading and $\alpha(t)$ is a blending weight between 0 and 1. Blending $\psi_{FF}(t)$ and $\psi_{FB}(t)$ provides the flexibility for maximizing the heading prediction accuracy in different conditions. For example, when the steering model does not have very high fidelity, $\alpha(t)$ can be chosen to be closer to 1 so that the prediction is less sensitive to modeling errors that resides in the *feedforward branch*. On the contrary, when predictor has limited bandwidth to predict a

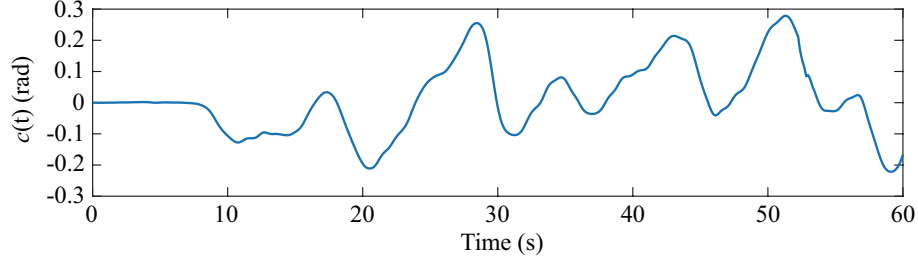


Figure 4.3: Coupling errors of vehicle heading based on one set of human driving data in teleoperated vehicles.

heading signal of high frequencies under large delays, $\alpha(t)$ can be chosen to be closer to 0, so that the heading prediction depends less on the Vehicle Predictor output. Details of the model-free predictors in *feedback branch* and the model-based steering feedforward loop are explained, respectively, preparing an open-loop test to evaluate the heading prediction accuracy using the blended architecture.

4.2 Predictor Design

Both the Driver Predictor and Vehicle Predictor in Figure 4.1 and Figure 4.2 have the same form as introduced in Chapter III and each signal in the input vector $\mathbf{y}_i(t)$ ($i \in \{1, 2\}$) to either predictor is predicted independently. Thus, predictor parameter λ to predict each signal is selected individually based on the predictor design procedure in Section 3.7. In this section, an example of predicting the signal of vehicle heading using the Vehicle Predictor is illustrated to show the design procedure in detail. Note that based on the blended architecture in Figure 4.2, the predictor heading output from the Vehicle Predictor is the output of the *feedback branch* and would be the feedback heading $\psi_{FB}(t)$ that contributes to the linear blending.

The heading signal to predict comes from the same data used in Section 3.6. There exists constant delays of 0.6 s in the signals as the inputs of the Vehicle Predictor. The coupling error $c(t)$ between undelayed heading $\psi_u(t)$ and delayed heading $\psi_d(t) = \psi(t - 0.6)$ is shown in Figure 4.3. A coupling error on the order of 5 deg

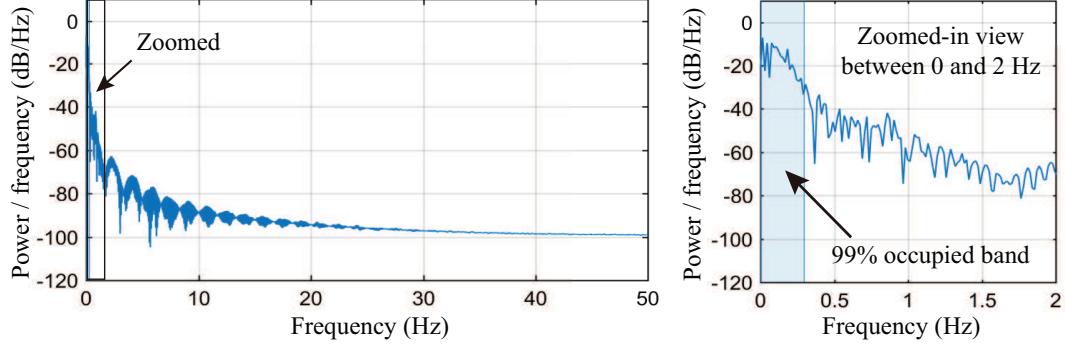


Figure 4.4: The power spectral density (PSD) of $c(t)$ with zoomed-in view. The 99% occupied band is between 0.005 Hz and 0.296 Hz.

corresponds to the difference between the actual undelayed heading of the remote vehicle and delayed heading perceived by human, causing a significant degradation in teleoperation performance: Track keeping error and steering control effort are increased by 76% and 18% respectively when compared to the metrics based on the same operator's data in the case without any delay (i.e. zero coupling error).

Define a metric $(\Delta\psi)_0$ as the 2-norm of coupling error for a total length of 60 s:

$$(\Delta\psi)_0 := \|\psi_a(t) - \psi_u(t)\|_2 \quad (4.2)$$

$(\Delta\psi)_0 = 8.88$ rad/s, which quantifies how much delays of 0.6 s distorts $\psi_u(t)$. $(\Delta\psi)_0$ is also considered as the baseline without prediction and is compared to the metric $(\Delta\psi)_b$ calculated based on the predictor output in (4.3) in this section.

With known delays $\tau = 0.6$, according to the design procedure, the first step is to determine bandwidth of the calculated $c(t)$ and range of ω_p . Since the data recorded and hence calculated $c(t)$ are in 100 Hz, the power spectral density of $c(t)$ based on a rectangular window for ω up to its Nyquist frequency of 50 Hz is shown in Figure 4.4, with the zoomed-in view on the right between 0 and 2 Hz. Note that majority of the signal power is distributed at low frequencies and the frequency band occupying 99% of the total integrated power of the spectrum is between 0.005 Hz and 0.296 Hz.

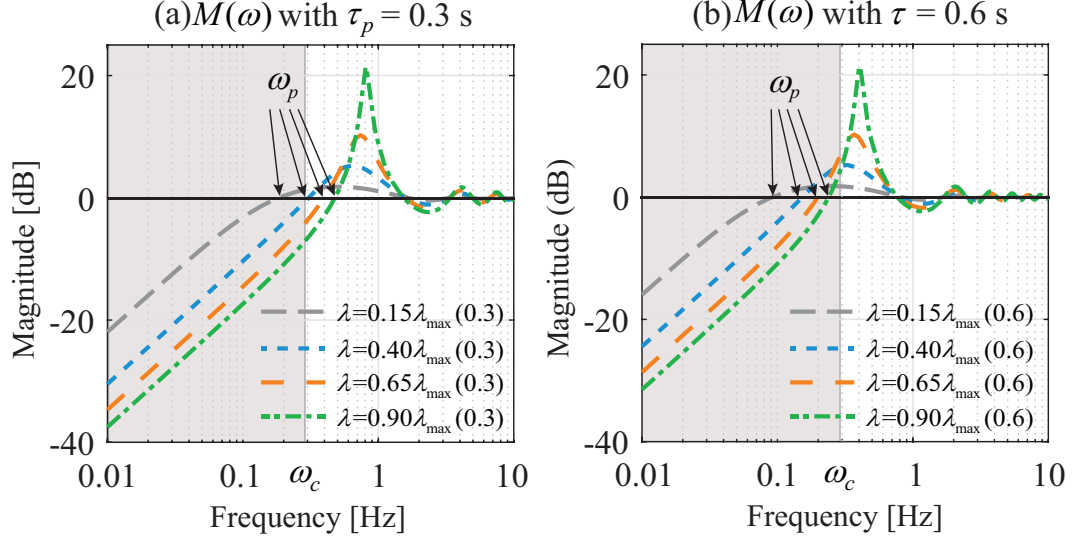


Figure 4.5: $M(\omega)$ with delays (a) $\tau_p = 0.3$ s and (b) $\tau = 0.6$ s. Frequencies within the bandwidth of $c(t)$, i.e. ω_c , are marked. Predictor improves steady state performance over frequencies with $M(\omega)$ less than 0 dB.

Thus, ω_c is estimated to be 0.296 Hz. Based on (3.26), the maximum ω_p is 1.60 rad (0.255 Hz) for delays of 0.6 s and predictor is always effective in attenuating $c(t)$.

In the second step, a range of λ to ensure predictor stability and consider good steady state predictor performance is identified. Since even the maximum ω_p cannot cover all $\omega \in [0, \omega_c)$, either Strategy 1 or Strategy 2 is applied to determine λ .

In Strategy 1, predictor can be designed to compensate partial delay $\tau_p = 0.3$ s and the ratio between $e(t)$ and $c(t)$, i.e. $M(\omega) = \left| \frac{e(t)}{c(t)} \right|$, with various λ smaller than the maximum value $\lambda_{\max}(0.3)$ for a stable predictor is plotted in Figure 4.5(a), along with the highlighted frequency range within ω_c of 0.296 Hz. Note that when λ is greater than $0.40\lambda_{\max}(0.3)$, $\omega_p > \omega_c$ is achieved. Thus, for Strategy 1, λ is lower-limited by $0.40\lambda_{\max}(0.3)$ in this step.

On the contrary, in Strategy 2, the actual amount of delays ($\tau = 0.6$) is compensated in full. $M(\omega)$ with various λ smaller than the maximum value $\lambda_{\max}(0.6)$ for a stable predictor is plotted in Figure 4.5(b). Using this strategy, λ is suggested to be less than $0.40\lambda_{\max}(0.6)$ so that for $\omega \in [0, \omega_c)$, a balance in steady state perfor-

mance between attenuating $c(t)$ when $\omega < \omega_p$ and amplifying $c(t)$ less when $\omega > \omega_p$ is achieved.

In the final step, predictors under various settings based on different λ specified in the second step and whether to include saturation and resetting scheme are simulated for the same heading signal input. Define a prediction accuracy metric $(\Delta\psi)_{\text{FB}}$ as the 2-norm of the prediction errors between undelayed heading $\psi_u(t)$ and predictor heading output $\psi_{FB}(t)$ for a total length of 60 s:

$$(\Delta\psi)_{\text{FB}} := \|\psi_{FB}(t) - \psi_u(t)\|_2 \quad (4.3)$$

$(\Delta\psi)_{\text{FB}}$ under different predictor settings are listed in Table 4.1. A smaller metric indicates more accurate prediction. Compared to without predictor ($(\Delta\psi)_{\text{FB}} = 8.88$ rad), predictors in all the settings generate outputs that are closer to the undelayed heading. Generally, larger λ leads to more accurate prediction. Applying saturation and resetting scheme helps reduce the overshoot in transient and also improves accuracy, except when $\lambda = 0.90\lambda_{\text{max}}(0.3)$ using Strategy 1. The reason of worse metric using saturation and resetting scheme for $\lambda = 0.90\lambda_{\text{max}}(0.3)$ is that with large λ , the predictor output after including the saturation resetting scheme is mostly saturated to a signal that is smooth but deviates from undelayed heading more at steady state.

Using Strategy 1, the predictor with $\lambda = 0.90\lambda_{\text{max}}(0.3)$ has the smallest metric of 4.70 rad; Using Strategy 2, the predictor with $\lambda = 0.40\lambda_{\text{max}}(0.6)$ and saturation and resetting scheme has the smallest metric of 4.02 rad, with a deduction of 55% compared to that without predictor. However, the best tuned predictor settings under both strategies are considered rather than preferring Strategy 2 over Strategy 1 due to the smaller metric. It is because in the proposed prediction framework, the feedback heading $\psi_{FB}(t)$ from the Vehicle Predictor will be blended with the feedforward heading $\psi_{FF}(t)$ and the resulting blended heading $\psi_b(t)$ is the final predicted

Table 4.1: Prediction accuracy metric $(\Delta\psi)_{\text{FB}}$ with various predictor settings compared to the baseline accuracy $(\Delta\psi)_0$

Without Predictor		$(\Delta\psi)_0 = 8.88 \text{ rad}$
Predictor settings		$(\Delta\psi)_{\text{FB}} \text{ (rad)}$
With Predictor (Strategy 1)	$\lambda = 0.40\lambda_{\text{max}}(0.3)$	5.11
	$\lambda = 0.40\lambda_{\text{max}}(0.3)$, with Sat. & Reset.	5.03
	$\lambda = 0.90\lambda_{\text{max}}(0.3)$	4.70
	$\lambda = 0.90\lambda_{\text{max}}(0.3)$, with Sat. & Reset.	5.84
With Predictor (Strategy 2)	$\lambda = 0.15\lambda_{\text{max}}(0.6)$	7.46
	$\lambda = 0.15\lambda_{\text{max}}(0.6)$, with Sat. & Reset.	5.70
	$\lambda = 0.40\lambda_{\text{max}}(0.6)$	5.56
	$\lambda = 0.40\lambda_{\text{max}}(0.6)$, with Sat. & Reset.	4.02

output. Prediction accuracy under different predictor design strategies and blending configurations remains to be studied.

4.3 Feedforward Branch

In this section, a model-based *feedforward branch* is illustrated.

The *feedforward branch* is model based in the sense that it relies on a steering model to represent the remote vehicle lateral dynamics. The steering model is used to map the operator's steering command $\delta(t)$ directly to a vehicle heading at the Driver Station that is directly available. Since this branch does not wait for the vehicle heading transmitted from the remote side controlled by sequence of steering $\delta(t)$, this branch borrows the terminology of feedforward and the heading output of the branch is called feedforward heading $\psi_{FF}(t)$. A modeling method that requires minimal knowledge about the governing equation of the vehicle is employed in this work to align with the motivation of developing a delay compensation method with minimal system information to benefit from performance robustness to modeling errors.

The vehicle platform generating the heading signal is a full-size notional High Mobility Multipurpose Wheeled Vehicle (HMMWV) model with 14 DoF vehicle dy-

namics and Pacejka tire model developed in [112], and the powertrain model in [44] that combines a 6L V8 diesel engine map, a drivetrain model and an automatic transmission model. This vehicle platform is used for all the following simulation results in Chapter IV and V. A potential steering model to capture the yaw dynamics can be represented by a second-order transfer function [113], with input of steering angle $\delta(t)$ and output of yaw rate $r(t)$:

$$G_{\text{ST}}(s) = \frac{r(s)}{\delta(s)} = \frac{K(1 + s/d)}{(1 + s/a_1)(1 + s/a_2)} \quad (4.4)$$

where the parameters K, d, a_1, a_2 are all positive numbers and are functions of vehicle longitudinal speed u along with constant vehicle parameters, such as yaw moment of inertia, vehicle mass, cornering stiffness and vehicle dimensions. Note that this model is deduced based on a bicycle model assuming small steering angles and constant tire cornering stiffness. However, due to vertical load transfer on tires and large steering angles for the vehicle platform of interest with high DoF and nonlinear tire models, K, d, a_1, a_2 may vary with different steering angle magnitudes. One way to establish (4.4) is to estimate or measure all the vehicle parameters needed, but obtaining accurate parameters could be difficult, especially when dealing with actual vehicles with their complicated dynamics. Instead, the modeling method used in this paper is to estimate K, d, a_1, a_2 directly through matching the frequency response of (4.4) with the vehicle platform. Specifically, sinusoidal steering angle profiles with amplitude A and frequency ω , i.e. $\delta(t) = A \sin(\omega t)$, are applied to the vehicle at constant longitudinal speed u and the yaw rate outputs $r(t)$ are recorded. When the vehicle reaches steady state, the gain and phase for the frequency response of (4.4) at the given frequency are then calculated as the ratio in magnitude and shift between the adjacent peaks of $\delta(t)$ and $r(t)$. The vehicle is simulated for various A and u , with A ranging from 0.02 rad to 0.10 rad with an increment of 0.02 rad, and u ranging

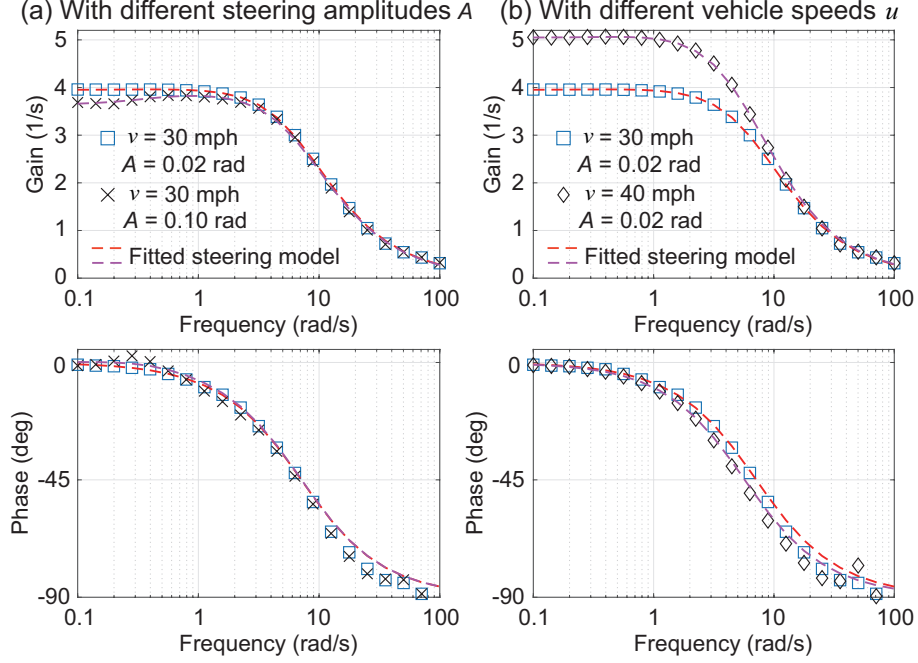


Figure 4.6: Gains and phases of the frequency response of (4.4) (a) with same u and different A (b) with same A and different u . The fitted steering model matches with collected data points well for each pair of $\{A, u\}$.

between 5 mph to 50 mph with an increment of 5 mph. For each pair of $\{A, u\}$, the data points of gain and phase from frequency of 0.1 rad/s to 100 rad/s are collected. Figure 4.6 shows the data points of gain and phase with different A and u . In Figure 4.6(a), with the same u of 30 mph, data points of phase are very close with A of 0.02 rad and 0.10 rad, but gains differ especially at low frequencies. In Figure 4.6(b), with the same A of 0.02 rad, higher u results in larger gains.

Steering model parameters are determined using the robust Least Absolute Residuals (LAR) method with MATLAB[®] Curve Fitting Toolbox[™] [114]. Starting with the condition of $A = 0.02$ rad for each u , data points of gain are fitted to the gain function of (4.4) between ω of 0.1 rad/s and 100 rad/s to obtain K, d, a_1, a_2 :

$$\text{Gain} = |G_{\text{ST}}(j\omega)| = K \sqrt{\frac{1 + (\omega/d)^2}{(1 + (\omega/a_1)^2)(1 + (\omega/a_2)^2)}} \quad (4.5)$$

With larger A , both a_1 and a_2 are constrained to be the same as the values obtained when $A = 0.02$ rad and the remaining two parameters K and d are refitted only to avoid the unnecessary changes of all the parameters for different A when the vehicle speed remains constant. In Figure 4.6, the steering model is fitted well to the data collected using the simulated vehicle platform and thus it can be used to at least represent the steady state lateral dynamics of a 14 DoF vehicle over the low frequencies. The fitted parameters are stored as a 2D lookup table for pairs of $\{A, u\}$.

For a given steering angle magnitude A and vehicle speed u within the fitting range, yaw rate is estimated using the steering model (4.4) with parameters in the lookup table, $K(A, u), d(A, u), a_1(u), a_2(u)$. Instead of implementing another model to capture the vehicle's longitudinal speed u without delays in the *feedforward branch*, the predicted $u_p(t)$ from the Vehicle Predictor outputs is utilized. Thus, the steering feedforward branch is coupled with the speed prediction as shown in Figure 4.2. The feedforward heading $\psi_{FF}(t)$ is then determined by integrating the yaw rate output of the steering model and is expressed in the form of $\psi_{FF}(t) = \psi_{FF}(t, u_p(t), \delta(t))$.

4.4 Open-Loop Evaluation of Heading Prediction Accuracy

An open-loop test is performed in simulation to evaluate the heading prediction accuracy of the proposed blended architecture in the predictor-based framework (Figure 4.1), compared to using either the model-free predictors or *feedforward branch* alone.

The test setup is shown in Figure 4.7. There exist control delays of 0.3 s and sensor delays of 0.6 s. Control commands $\mathbf{y}_1(t)$ including throttle $Th(t)$, brake $Br(t)$ and steering $\delta(t)$ from the same data used in Section 4.2 are the inputs to the whole system and the output of interest is the blended vehicle heading $\psi_b(t)$. $\psi_u(t)$ and $\psi_d(t)$ represent the undelayed and delayed heading, respectively. Prediction accuracy is evaluated by comparing predicted $\psi_b(t)$ to the baseline of $\psi_u(t)$ and a metric of quantifying the accuracy, $(\Delta\psi)_b$, is calculated as the L_2 norm of the heading errors

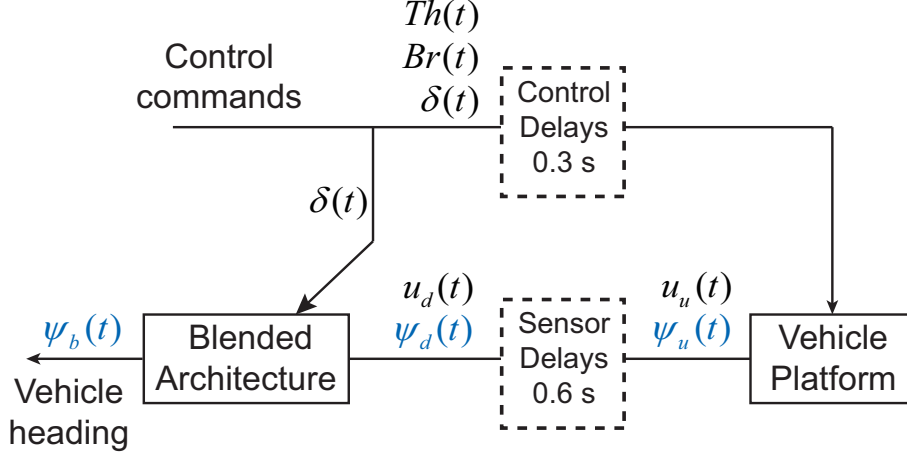


Figure 4.7: An open-loop test setup to evaluate the heading prediction accuracy of the blended architecture.

between $\psi_b(t)$ and $\psi_u(t)$:

$$(\Delta\psi)_b := \|\psi_b(t) - \psi_u(t)\|_2 \quad (4.6)$$

Smaller $(\Delta\psi)_b$ indicates better heading prediction accuracy.

Note that the Driver Predictor irrelevant to heading prediction is not included and only the prediction of heading as well as speed in the Vehicle Predictor within the blended architecture are of interest in this test. λ values for predictor heading $\psi_{FB}(t)$ and predictor speed $u_p(t)$ are selected based on the procedure described in Section 3.7 and listed in Table 4.2. Note that two kinds of Vehicle Predictor are tested and are referred as Predictor I and II, respectively. They have the same λ for speed prediction, but different λ for heading prediction that correspond to the ones with the best prediction accuracy highlighted in Table 4.1 (i.e., $\lambda = 0.90\lambda_{\max}(0.3)$, and $\lambda = 0.40\lambda_{\max}(0.6)$, with Sat. & Reset.). Two different configurations of blending the outputs from the *feedforward branch* and *feedback branch* are proposed and listed in Table 4.2, when dealing with Predictor I or Predictor II and are explained next.

In Configuration I, *feedback branch* includes Predictor I designed using Strategy

Table 4.2: Two configurations that blend *feedforward branch* and *feedback branch* to predict vehicle heading

Blending Configuration I				
<i>Feedforward branch</i>	$\psi_{FF}(t) = \psi_{FF}(t, u_p(t), \delta(t))$			
<i>Feedback branch</i>	Vehicle Predictor I			
	Signal	Strategy	Parameter λ	Sat. & Reset.
	$\psi_{FB}(t)$	Strategy 1	$0.90\lambda_{\max}(0.3)$	No
	$u_p(t)$	Strategy 2	$0.10\lambda_{\max}(0.6)$	Yes

Blending Configuration II				
<i>Feedforward branch</i>	$\psi'_{FF}(t) = \psi_{FF}(t, u_p(t), \delta(t - 0.3))$			
<i>Feedback branch</i>	Vehicle Predictor II			
	Signal	Strategy	Parameter λ	Sat. & Reset.
	$\psi_{FB}(t)$	Strategy 2	$0.40\lambda_{\max}(0.6)$	Yes
	$u_p(t)$	Strategy 2	$0.10\lambda_{\max}(0.6)$	Yes

1. Recall from Section 4.2 that Predictor I using Strategy 1 is effective in compensating partial delays of around 0.3 s and there remains a lag of around 0.3 s at best compared to $\psi_u(t)$. Also note that the feedforward heading $\psi_{FF}(t)$ represents the undelayed heading $\psi_u(t)$ with a time lead equal to the control delay of 0.3 s, because it is calculated at the Driver Station locally without going through the Vehicle Platform and therefore not experiencing the 0.3 s control delay that the actual steering command experiences. Thus, in Configuration I, blending the predictor output with a residual 0.3 s delay with the feedforward heading $\psi_{FF}(t, u_p(t), \delta(t))$ that has a time lead of 0.3 s directly is potentially beneficial to generate a blended heading close to $\psi_u(t)$.

In Configuration II, Predictor II using Strategy 2 in the *feedback branch* aims to compensate all the sensor delays of 0.6 s and there is no residual lag to $\psi_u(t)$. In this case, steering input $\delta(t)$ to the steering model is delayed by the same amount 0.3 s as the control delays to generate a postponed feedforward heading $\psi'_{FF}(t) = \psi_{FF}(t, u_p(t), \delta(t - 0.3))$ so that both branch outputs $\psi_{FB}(t)$ and $\psi'_{FF}(t)$ are the heading prediction at the same instance as $\psi_u(t)$.

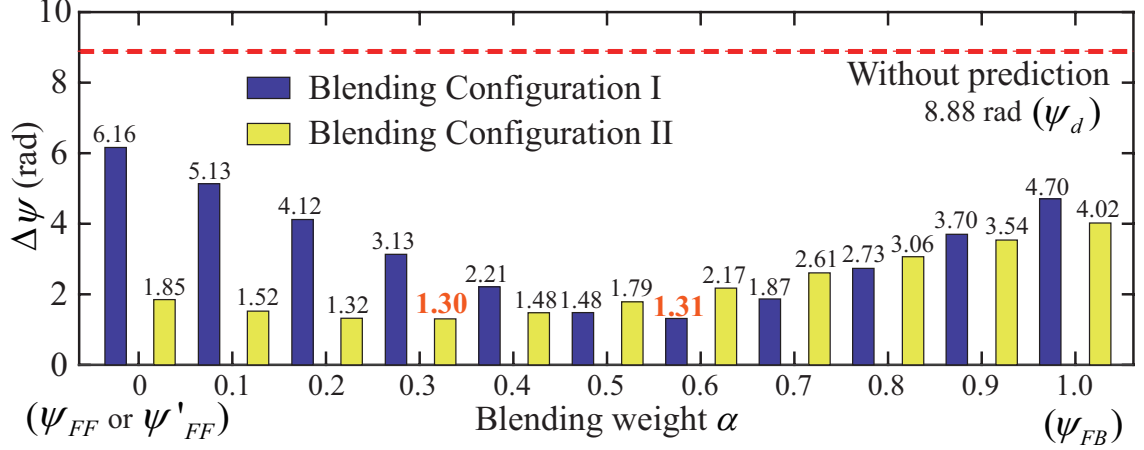


Figure 4.8: Prediction accuracy metric $(\Delta\psi)_b$ with various α . Compared to without prediction ($(\Delta\psi)_0 = 8.88$ rad), blended headings with $\alpha = 0.6$ using blending configuration I and with $\alpha = 0.3$ using blending configuration II have the best prediction accuracy of $(\Delta\psi)_b = 1.31$ rad and $(\Delta\psi)_b = 1.30$ rad, respectively.

The system in Figure 4.7 is simulated for 60 s with the blended architecture using these two Blending Configurations for different constant blending weights α ranging from 0 to 1 with an increment of 0.1. $(\Delta\psi)_b$ values with various α and the two blending configurations based on different predictor design strategies are summarized in Figure 4.8. Recall from Table 4.1 that without prediction, $(\Delta\psi)_0$ norm of errors between the data $\psi_u(t)$ and $\psi_d(t)$ is 8.88 rad. When $\alpha = 1$, $\psi_b(t)$ is the predicted output of the well designed predictor as in the example in Section 4.2, hence the corresponding metrics $(\Delta\psi)_b$ are the same as $(\Delta\psi)_{FB}$ reported in Table 4.1. When $\alpha = 0$, $\psi_b(t)$ is either $\psi_{FF}(t)$ or $\psi'_{FF}(t)$ based on corresponding configurations. For all α , $(\Delta\psi)_b$ is always smaller than $(\Delta\psi)_0$, indicating improved accuracy using the blended architecture to predict heading. Minimum $(\Delta\psi)_b$ and thus best prediction accuracy is achieved when $\alpha = 0.6$ for Blending Configuration I ($(\Delta\psi)_b = 1.31$ rad) and $\alpha = 0.3$ for Blending Configuration II ($(\Delta\psi)_b = 1.30$ rad), both resulting in a 85% improvement compared to without prediction. It is noteworthy that although $(\Delta\psi)_b$ using the postponed heading only is as low as 1.85 rad, the blended heading can reduce the metric $(\Delta\psi)_b$ even further down to 1.30 rad. Thus, blending can offer

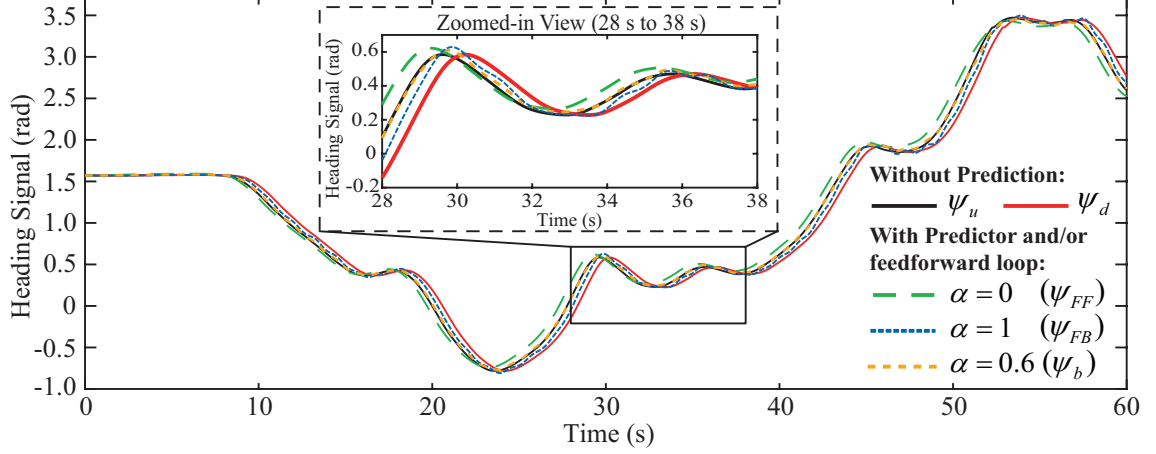


Figure 4.9: The open-loop output of the blended heading $\psi_b(t)$ with $\alpha = 0.6$ is very close to undelayed heading $\psi_u(t)$, indicating better prediction accuracy compared to that of either feedforward heading $\psi_{FF}(t)$ or feedback heading $\psi_{FB}(t)$ alone.

a more robust improved heading prediction accuracy.

Figure 4.9 shows the comparison of different heading signals based on blending configuration I. The delayed heading without prediction, $\psi_d(t)$, has a consistent delay of 0.6 s. Based on the curve $\psi_{FB}(t)$, the predictor is effective in compensating delays partially for around 0.3 s. The green dashed line is the feedforward heading $\psi_{FF}(t)$ and has, as mentioned earlier, a lead time of 0.3 s due to not being exposed to the control delay, leading to a $(\Delta\psi)_{FF}$ for $\alpha = 0$ as large as 6.16 rad. The blended predicted heading with $\alpha = 0.6$ is very close to $\psi_u(t)$ and outperforms both using $\psi_{FF}(t)$ or $\psi_{FB}(t)$ alone.

In summary, this open-loop test shows that the blended architecture can achieve a more accurate prediction of vehicle heading than using either the model-based *feedforward branch* or the model-free *feedback branch* alone.

CHAPTER V

Simulation Based Human-In-the-Loop Test

In this chapter, the effectiveness of the developed predictor based framework is evaluated through human-in-the-loop experiments in a simulation platform. The test scenario is that human drivers control a simulated vehicle in a virtual environment in order to complete a virtual track following teleoperated driving task. How much the predictor based framework can improve the performance of vehicle mobility and drivability under large round trip delays are presented. The work in this chapter is based on publications [111, 115].

5.1 Simulation Platform

To evaluate the predictor based framework experimentally, a real-time human-in-the-loop simulation platform has been developed in MATLAB[®] Simulink[®] to emulate a teleoperated vehicle system as shown in Figure 5.1.

5.1.1 Vehicle Platform

At the vehicle side, a typical military truck, namely, a notional High Mobility Multipurpose Wheeled Vehicle (HMMWV), is chosen as the vehicle platform and simulated using a model developed in [112, 44], capturing the 14 DoF vehicle dynamics, tire forces and powertrain dynamics. Note that this model is the same as the one

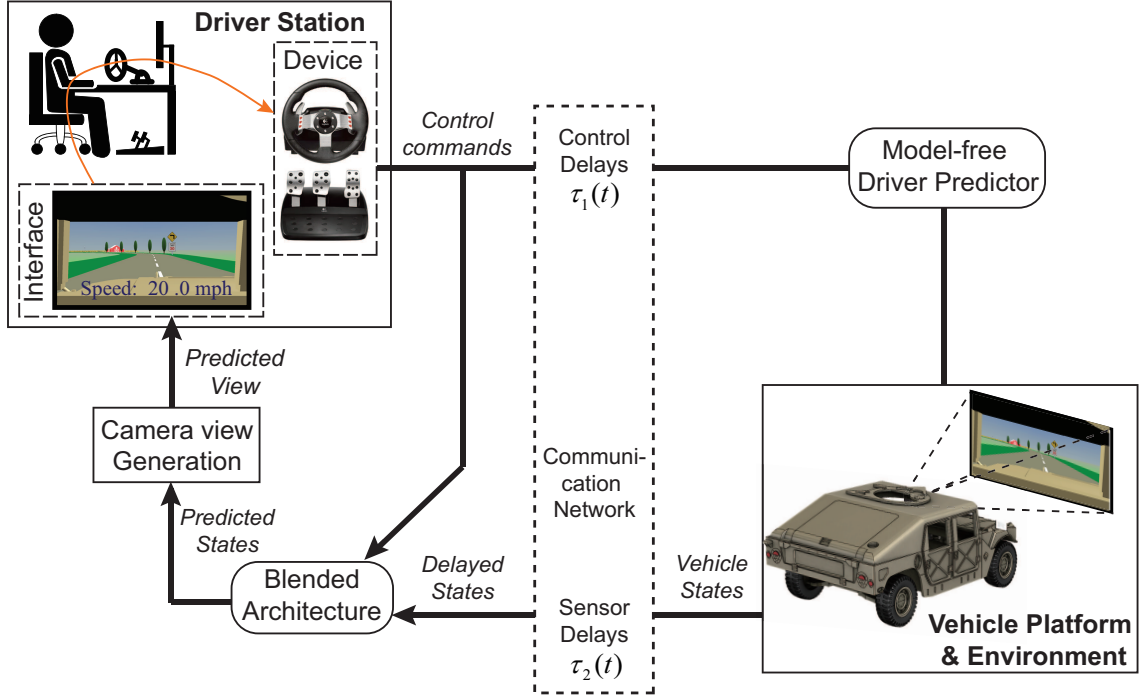


Figure 5.1: Human-in-the-loop simulation platform with predictor based framework. The vehicle model is a 14 DoF model of a typical military truck. By means of a steering wheel and set of pedals, a human can control the throttle, brake and steering to drive the vehicle. Control and sensor delays can be specified independently. The prediction framework and blended prediction architecture can be activated or deactivated.

used in Section 4.3 to fit a steering model. The vehicle is animated in the virtual environment using the MATLAB[®] Simulink[®] 3D Animation Toolbox[™] [116] and moved around by specifying its global position and heading direction. A camera with horizontal field of view (FOV) of 55 deg and vertical FOV of 33 deg is placed above the driver’s seat within the vehicle to provide the first-person camera view. Note that in this simulated environment, the camera view is not transmitted as video frames in the simulation platform. Instead, it is directly available from the virtual environment by changing the position and heading of the vehicle (i.e. where the camera is located) based on provided states.

5.1.2 Driver Station

The driver station contains a monitor updating a driving interface at 20 fps, along with a set of Logitech G27 steering wheel and pedals for lateral and longitudinal control of the vehicle. Figure 5.2 shows the driving interface with a resolution of 1420x800 pixels (aspect ratio of 16:9), in which the vehicle speed is overlayed on the camera view. The camera is zoomed and the environment is visible through the vehicle windshield at the driver’s side. A hook on the vehicle hood is shown as a reference of vehicle’s center, i.e., if the hook aligns with the closest centerline strip and the vehicle is driven in a straight line, the vehicle center is on the centerline.

The physical steering wheel and pedals are connected as a joystick to the computer that runs the simulation platform. A Simulink[®] block named “Joystick Input” captures the encoder reading of the steering wheel and positions of throttle and brake pedals, and the steering angle as well as throttle and brake commands are generated by scaling these readings.

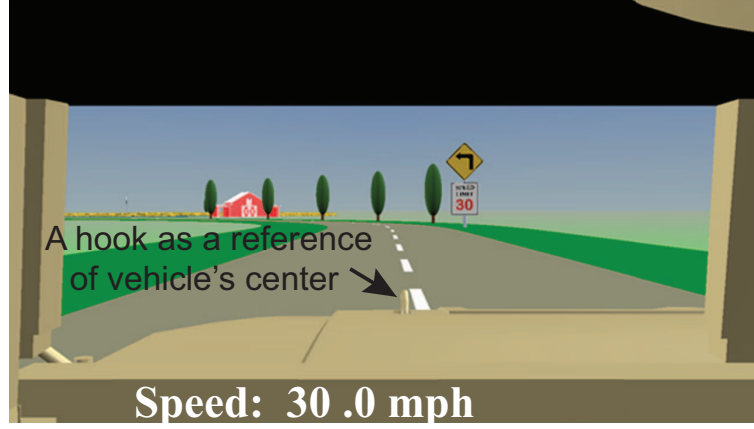


Figure 5.2: The driving interface displays the camera view and current vehicle speed. A hook on the vehicle hood is considered as a reference of the vehicle's center.

5.1.3 Platform Capability

In the closed-loop simulation, the human uses the physical steering wheel and pedals to control vehicle's heading and speed based on the visual feedback. Thus, the loop in the teleoperated vehicle system is closed by the human driver as a controller. The signals transmitted between the Driver side and the Vehicle side are control commands (including steering, throttle and brake) and vehicle states (including heading angle, global position and vehicle speed). The derivative of the signals and the send time are also included in the packets for transmission, allowing the predictors to perform prediction. While the derivative of all the vehicle states are directly available within the simulated vehicle model, numerical differentiations are performed on the control commands, since only the signal measurements are available. Also, due to the observation that in a majority of normal driving scenarios the steering commands generated by a human do not exceed a frequency of 1 Hz [117, 118], a low pass filter with cutoff frequency of 1 Hz is implemented on the steering signals. Throttle and brake signal are saturated between 0 and 1 to generate feasible commands for vehicle powertrain dynamics.

Note that this simulation platform is not set for distributed simulation, meaning

that both the Driver side and Vehicle side are simulated in the same Simulink file on one computer and signals are transmitted directly without a communication network. Thus, virtual delays (either constant or varying delays) are optionally added to simulate the situation of a teleoperated vehicle system with some delays of interest. Virtual delays can be added in two ways, either as a predefined time sequence or as a random process. Delays in either direction between the Driver side and Vehicle side, i.e. control delays $\tau_1(t)$ and sensor delays $\tau_2(t)$, can be specified independently.

The simulation platform also has the general option to activate or deactivate the predictor based framework without any structure change. When λ in the Driver Predictor and Vehicle Predictor is set to zero, there will be no prediction of control commands and vehicle states using model-free predictors. Also, α can be specified as different values to generate different blended heading of interest: when $\alpha = 1$, blended heading completely relies on the predictor outputs and thus the predictor based framework is model-free to preserve robustness to modeling error. When $\alpha \in (0, 1)$, blended heading is a linear combination of outputs from the Vehicle Predictor and steering feedforward loop and could improve prediction performance of vehicle heading even more than the former case of using Vehicle Predictor only ($\alpha = 1$) (referring to the open loop test result in Chapter IV).

5.2 Experiment Design

Using the simulation platform described above, human-in-the-loop experiments were performed. In this section, details of the experiment design are presented.

5.2.1 Simulated delays

Recall in Chapter II that a round trip delay of interest is between 0.3 s and 1.0 s so that human drivers can control the vehicle with pure teleoperation and delays will significantly affect the vehicle mobility. In this experiment, a round trip delay of

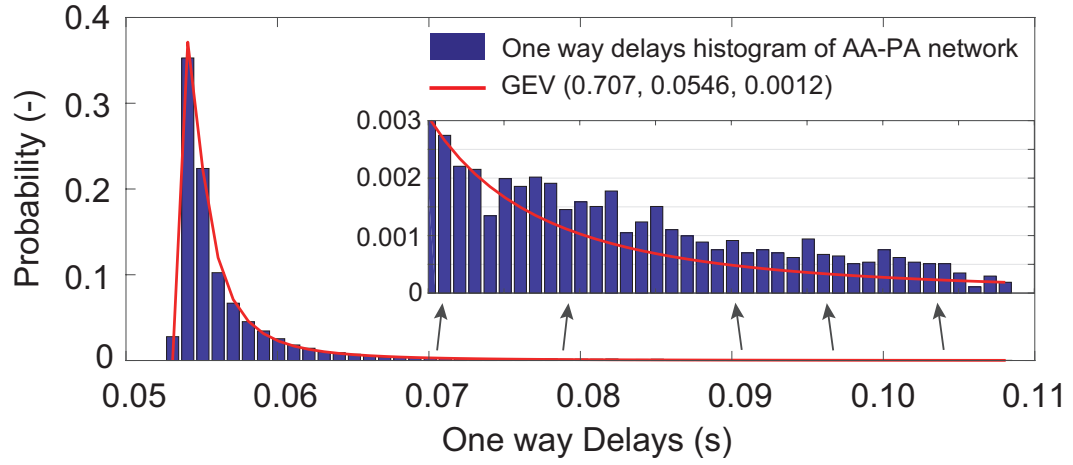


Figure 5.3: The histogram of one way delay measurement data of MI-CA network. It can be fitted using a Generalized Extreme Value distribution with $k = 0.707$, $\mu = 0.0546$, $\sigma = 0.0012$.

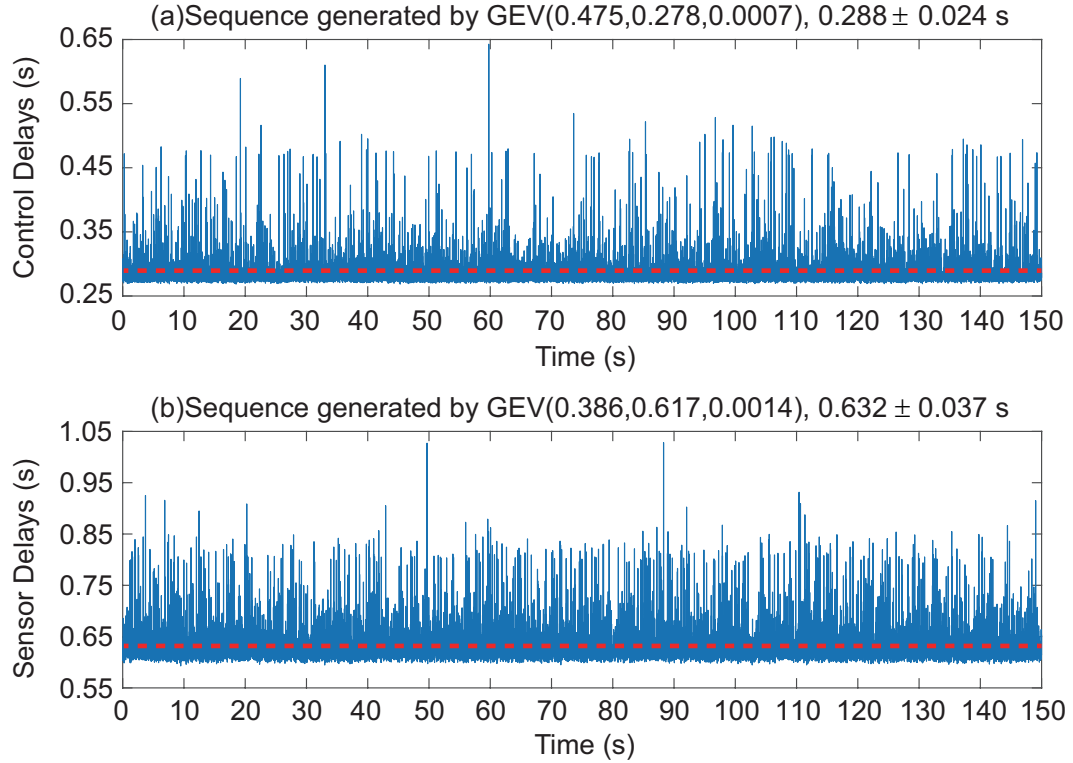


Figure 5.4: Simulated delay sequences of (a) control delays and (b) sensor delays, generated by the corresponding fitted GEV models. Mean values are marked as red lines.

around 0.9 s is tested. Specifically, the delay is distributed in two ways with control delays of around 0.3 s and sensor delays of around 0.6 s based on the assumption that sensor delays are larger than control delays.

For the constant delay case, transmitted signals in two directions (from Driver to Vehicle and from Vehicle to Driver) are consistently delayed by 0.3 s and 0.6 s, respectively.

For the varying delay case, two varying delay sequences with mean delay values matching the value of 0.3 s and 0.6 s of the constant delay case are generated based on the actual network data collected in [46] between Ann Arbor (AA), Michigan and Palo Alto (PA), California. Note that the same set of data has been used in this work in Section 3.2.2 and the histogram of round trip measurement of this AA-PA network is shown in Figure 3.5. Assuming delays in both ways are the same, Figure 5.3 shows the histogram of one-way delays. The mean and standard deviation are 0.058 s and 0.006 s. There exist several observations about the shape of the histogram, mostly due to the fact that large delay values are measured when there exist frequent spikes with large magnitude. First, unlike a normal distribution in a symmetric bell shape, the histogram is right-skewed. Second, consider the region where data is far from its mean value as the tail. The tail is heavy (i.e., 2.3% data are not within the range of three standard deviation from mean compared to 0.3% in standard normal distribution) and long (i.e., distributed up to 0.108 s). The right-skewness and heavy long tail observed in the histogram are expected for network data and match with the observations in [119].

A delay model is implemented to represent the one-way delays of AA-PA network for the purpose of generating additional delay sequences with similar characteristics to the measured data. One group of approaches of generating a delay model considers network delay as a random process and delays are captured based on Markov chain [120, 121] or autoregression and moving average (ARMA) models [122] and their

extensions. The transitions of delay values in successive time are included in these models. Another group of approaches assumes that the random variable of discrete delay value at each time instance, denoted as τ , follows an independently and identically distributed (IID) distribution. Potential distributions to fit network delays in the literature include, but are not limited to, Pareto, gamma, lognormal, and Weibull distributions [119, 123, 124, 125, 126]. In terms of tail shape, it is discussed in [127] that Pareto distribution generally has the heaviest tail among all four and follows the lognormal distribution with moderate tail; Weibull distribution has a short tail when the magnitude of the peak is large (like the high relative probability in the histogram in Figure 5.3). However, the actual tail shape depends on the fitted parameter and it is hard to pick out the distribution that could lead to best fitting.

Given that, a Generalized Extreme Value (GEV) distribution is used to fit the data in this work. It provides the flexibility to control the shape and size of the tails using single shape parameter ξ . When $\xi > 0$, a heavy tailed distribution such as Pareto can be represented and has a lower bound based on the extreme value theory. When $\xi < 0$, GEV distribution has a short tail and is similar to reverse Weibull distribution with an upper bound. When $\xi = 0$, it can express gamma and lognormal distributions. Denote τ as the random variable of delay and define the standardized τ_0 as $\frac{\tau - \mu}{\sigma}$, where μ and σ are the location and scale parameters, respectively. Its probability density function is [128]:

$$\begin{aligned} f_{\xi \neq 0, \mu, \sigma} &= \frac{1}{\sigma} (1 + \xi \tau_0)^{-1-1/\xi} \exp(-(1 + \xi \tau_0)^{-1/\xi}) \text{ with } 1 + \xi \tau_0 > 0 \\ f_{\xi=0, \mu, \sigma} &= \frac{1}{\sigma} e^{-\tau_0} \exp(-\exp(-\tau_0)) \end{aligned} \quad (5.1)$$

Also, based on the extreme value theory, the bound of the random variable, τ_{bound} , is determined based on $\tau_0 = \frac{\tau - \mu}{\sigma}$:

$$\tau_{\text{bound}} = \mu - \frac{\sigma}{\xi} \quad (5.2)$$

Table 5.1: Simulated control and sensor delays generated by the fitted GEV models. The means are 0.288 s and 0.632 s, respectively.

	Fitted GEV			Mean (s)	Standard deviation (s)
	ξ	μ	σ		
Control delays	0.475	0.278	0.0007	0.288	0.024
Sensor delays	0.386	0.617	0.0014	0.632	0.037

which is the lower bound of delays when $\xi > 0$ and upper bound when $\xi < 0$.

Parameters of the GEV distribution are determined based on maximum likelihood fitting, and the resultant delay model for one-way delays of the AA-PA network is denoted as $\text{GEV}(\xi, \mu, \sigma)$, with parameters $\xi = 0.707, \mu = 0.0546, \sigma = 0.0012$. Note that $\xi > 0$, which confirms the observation of heavy long tail. In Figure 5.3, delay model $\text{GEV}(0.707, 0.0546, 0.0012)$ fits the histogram well. However, the mean value is only 0.058 s. To match with the values of 0.3 s and 0.6 s in the constant delay case, 5 and 11 independent randomly generated sequences using the delay model $\text{GEV}(0.707, 0.0546, 0.0012)$ are summed, respectively. The summed delay sequences are analogous to the delays measured when packets are transmitted in TCP and bounced back and forth for multiple times before marked as received at the other side in the AA-PA network. Similarly, the histograms of summed sequences are fitted using GEV distributions. Then new sets of delay sequences are randomly generated based on the fitted distributions for 150 s and are used as the simulated control and sensor delays in the tests, as shown in Figure 5.4. They also meet the assumption that delays can be modeled using a two-time-scale Markov Chain and thus the predictor stability with varying delay is established based on λ values in (3.20). The fitted parameters of GEV distributions as well as means and standard deviations of the simulated delays are listed in Table 5.1. Shape parameters of both models are greater than 0, indicating the existence of heavy tail in the simulated control and sensor delays sequence. The mean control and sensor delays are 0.288 s and 0.632 s, respectively, and thus for the varying delay case the round trip delays have a mean of 0.92 s.

Table 5.2: Different predictor settings to predict individual signal of interest

	$c(t)$		Predictor Settings			
Predicted Signal	Measured Delay τ_m [s]	ω_c [rad/s]	Predicted Delay τ_p [s]	λ [s^{-1}]	ω_p [rad/s]	Sat. & Reset.
Throttle	0.3	11.5	0.3 (Strategy 2)	$0.30\lambda_{\max}(\tau_p)$	1.65	No
Brake		19.4		$0.30\lambda_{\max}(\tau_p)$	1.65	No
Steering		5.37		$0.10\lambda_{\max}(\tau_p)$	0.94	Yes
Speed	0.6	3.35	0.3 (Strategy 1)	$0.40\lambda_{\max}(\tau_p)$	1.92	Yes
Global X		1.88				No
Global Y		1.30				No
Heading		1.86				Yes

5.2.2 Parameters in the Predictor Based Framework

Parameters in the predictor based framework to be designed include λ values in both Driver Predictor and Vehicle Predictor to predict each of the transmitted signals, and the blending weight α when combining the model-based *feedforward branch* and model-free *feedback branch* together.

All the λ values were selected based on the design procedure in Section 3.7 and are listed in Table 5.2. As mentioned in the procedure, some representative data is needed to estimate the coupling error of the signal to be predicted. In this experiment, preliminary data were collected from an expert human driver who has years of experience with teleoperated driving using the developed simulation platform under large delays. For each of the seven transmitted signals listed in Table 5.2, $c(t)$ between the undelayed and delayed signals was calculated and power spectral density of $c(t)$ was performed. The coupling error bandwidth was estimated by integrating the signal power over the frequency until 90% of the total signal power was reached. The bandwidth ω_c listed in the table is the average estimated value based on three sets of data from the same driver.

Note that ω_c for control commands of throttle, brake and steering are of large

magnitude and it is unlikely that the predictor bandwidth ω_p can cover all the frequencies up to ω_c . Thus, smaller λ were designed based on Strategy 2 (compensating same amount of delay as measured). Saturation and resetting scheme was included to aid with steering prediction, but including it for throttle and brake prediction was not critical, because these two commands were already saturated between 0 and 1 inherently.

On the contrary, ω_c for the vehicle states of speed, position and heading is relatively small due to the fact that the vehicle model with 14 DoF dynamics acts like interconnection of multiple low-pass filters. Partial delays of 0.3 s were compensated with λ selected for a large enough $\omega_p > \omega_c$. Since frequent direction changes in vehicle speed and heading were expected, saturation and resetting scheme was applied when predicting these two signals.

Finally, in the blended architecture, α was set to be 0.5 to blend the heading from the Vehicle Predictor output and the feedforward heading from the the steering feedforward loop with equal weighting as a first step, because there exists no direct evidence indicating that a tuning for best open-loop performance would also result in the best closed-loop performance.

5.2.3 Test Details

A track was generated in the virtual environment as shown in Figure 5.5. The track (in gray) is 810 meters long and 10 meters wide with a dashed white centerline. Shoulders of 6 meters width (in dark green) are located on either side of the track. At each turn, a safe speed was priorly determined as the maximum speed when the vehicle model can successfully be driven on the centerline without causing any tire lift-off and the value is shown on a traffic sign indicating the speed limits before each turn. Trees at the turns and other landmarks such as mountains, houses, cornfields and water served as visual cues to aid operators on speed and distance assessment.

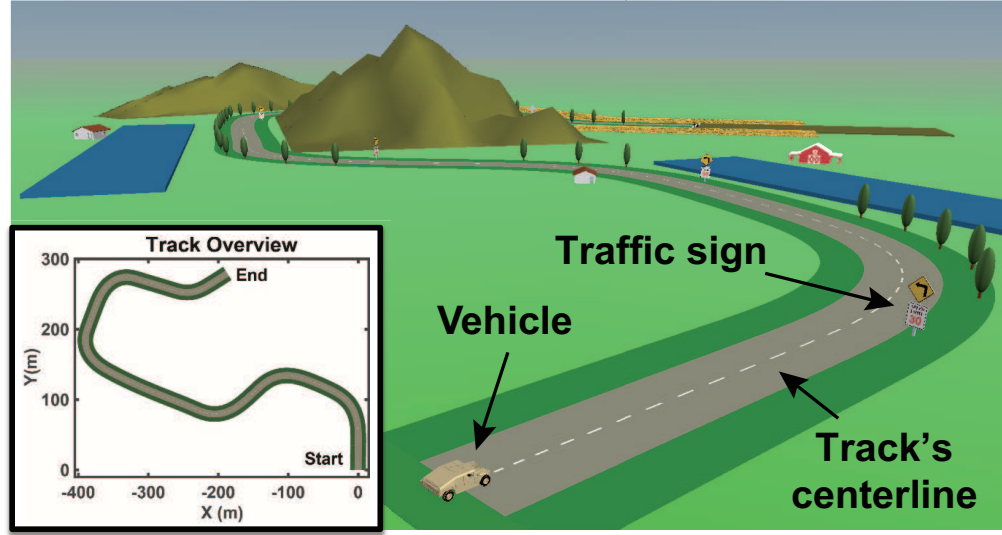


Figure 5.5: Designated track, vehicle and landmarks in the virtual environment.

The task for the human drivers was to control the vehicle to complete the track (by passing the endline of the track) as fast as possible and stay as close to the track's centerline as possible. Three performance metrics were considered as the dependent variables: track completion time, track keeping error and steering control effort. Track completion time captured how fast the vehicle was driven. Track keeping error was calculated by the area between the trajectory of the vehicle and track's centerline, indicating how close the vehicle followed the centerline. These two metrics reflected the longitudinal and lateral performance of the vehicle, respectively. Smaller metrics indicated better mobility. Steering control effort was determined by integrating the steering angle with respect to time and normalizing the integral with respect to the track completion time. This metric aimed to capture the drivability of the vehicle. The smaller the steering control effort was, the easier human drivers controlled the vehicle. These three metrics are referred to as *Time*, *Error* and *Effort*, respectively, in the following contents.

Two independent variables that could affect the dependent variables were studied: delay type and prediction method. Delay type could be chosen from $\{\mathbf{CD}, \mathbf{VD}\}$, where **CD** and **VD** refer to the constant delay case and varying delay case in the form of simulated sequences, respectively, as determined in Section 5.2.1. Prediction method was among $\{\mathbf{NoPred}, \mathbf{Pred}, \mathbf{PredBlend}\}$, where **NoPred** represented the case with non-zero delays, but without activating predictors and blended architecture. The only difference between method of **Pred** and **PredBlend** was the α value. For **Pred**, $\alpha = 1$ so that only the two model-free predictors were included in the closed-loop system, whereas for **PredBlend**, $\alpha = 0.5$ and a resultant blended heading prediction was generated with the predictors in addition to the *feedforward branch*. Thus, the experiment was a 2 (delay type with two levels) x 3 (prediction method with three levels) within-subjects factorial design. The goal of the experiment was to study how delay type and prediction method affected the performance of vehicle mobility and drivability in teleoperated vehicles operated at high speed.

Seven scenarios with three repetitions in each scenario were tested. The seven scenarios included a **NoDelay** scenario with zero delays used as the baseline performance of human drivers to handle this track-following teleoperated driving task, as well as the six scenarios with non-zero delays that are configured using combinations of delay type and prediction method. Thus, each subject performed a total of 21 runs. The order of the runs was randomized in an evenly distributed manner to reduce the learning factor of subjects on a single scenario. Specifically, no single scenario was tested twice in a row and there does not exist a situation when one scenario was tested two times more than another scenario.

5.2.4 Test Procedure

Test subjects were recruited through flyers and campus activity announcements at the University of Michigan. They were paid ten dollars for participating in and

completing the experiment. At the start of each user test, subjects filled out an informed consent form and answered some basic background questions about their age, proficiency of driving a vehicle in the real world, and proficiency of driving in a virtual environment with gaming steering wheel and pedals. The whole test process lasted no more than 2 hours and was divided into a training session (1 hour) and a testing session (45 minutes).

Recall that large delays could make teleoperated driving at high speed very challenging. Therefore, the training session was necessary to help subjects adapt to large delays and drive in different test scenarios. In the training session, subjects were verbally instructed first on the test details including the track-following driving task, three performance metrics, as well as seven scenarios to test. Subjects were informed that even though both metrics of completion time and track keeping error were studied and of equal importance, controlling the vehicle to stay on the track was the priority of the task and potentially reducing vehicle speed in exchange of ensuring it was encouraged. However, it was also mentioned to the subjects that when driven too slow, the run may not be counted as valid in the following testing session, because the experiment was to test vehicle mobility when the vehicle was operated at high speed. The remaining time in the training session was left for subjects to adapt to the teleoperated driving setup and practice all seven test scenarios one by one. The current scenario to practice was visually displayed on the driving interface and the three performance metrics were calculated each time the subjects completed the driving task to monitor their performance under training. A scenario would be practiced until metrics between the successive three trials indicated consistent performance and subjects were confident about driving in such scenario. Subjects also could ask for more time to be distributed on the scenarios which they were not familiar with.

In the testing session, each subject was asked to complete a total of 21 valid runs with randomized order. The run was considered as valid if none of the following

conditions were observed:

1. Vehicle was off track for more than 5 s or did not pass the endline of the track.
2. Two vehicle tire lift-offs were detected in the model.
3. Average vehicle speed was less than 25 mph.

The 25 mph speed bound was determined based on the driving data of the Beta Test. A human driver went through the training session and became well-trained to generate replicates of test run data with consistent performance. In the most challenging scenarios with varying delay and without prediction (i.e. **VD x NoPred**), the expert driver was able to complete the task within 60 s and stay on the track all the time. The resultant average vehicle speed was around 30 mph. A 5 mph reduction compared to the Beta Test was allowed and therefore 25 mph was set to be the lowest qualifying average speed. A run with average speed lower than it meant that subjects did not push the vehicle dynamics to the limit to complete the driving task in this run.

After all test runs were completed, subjects were thanked for their participation and dismissed.

5.3 Analysis Methods

2-way repeated measures analyses of variances (RM-ANOVA) were used to study the influence of the two independent variables of delay type and prediction method on each of the dependent variables, i.e. performance metric *Time*, *Error* and *Effort*. Data was analyzed using the software Minitab 17. Applying RM-ANOVA requires that data meet two assumptions: normality and sphericity. For each metric, normality was checked using the Anderson-Darling test and sphericity was checked by comparing variances across the six test conditions of (delay type) x (prediction method). Data

runs whose metrics were not within three standard deviations from the mean of the data worth further investigation and potential processing such as performing log scale transformation on data.

Two null hypotheses for each metric was tested using an F test based on Type III sums of squares:

1. There does not exist a significant difference in performance metrics when different prediction methods are applied to compensate delays.
2. There does not exist a significant difference in performance metrics between the constant and varying delays used for testing.

If the F test provided evidence ($p < 0.05$) that at least one mean is different from the rest, Fisher's least significant difference (LSD) method was used to identify the groups with pairwise mean differences that were significant ($p < 0.05$).

5.4 Results

The experiments were approved by the University of Michigan Health Sciences and Behavioral Sciences Institutional Review Board (UM IRB #HUM00112376). A total of 22 test subjects participated in the experiments. 2 of them were observed to still struggle with completing the driving task especially in the conditions without any prediction after 1 hour of training session. The background information collected revealed that they reported to have only 1-6 months of experience on driving vehicles in the real world and no experience of driving virtual vehicles at all. Another subject did not push the vehicle to its limits and the calculated metrics were far from those of the rest subjects' data. Besides, data of these three subjects violated the normality test and were considered as outliers. Therefore, they are not included in the analysis.

The remaining 19 subjects had an average age of 22.7 years with standard deviation of 2.6 years. Majority of them were familiar with doing daily driving in the real

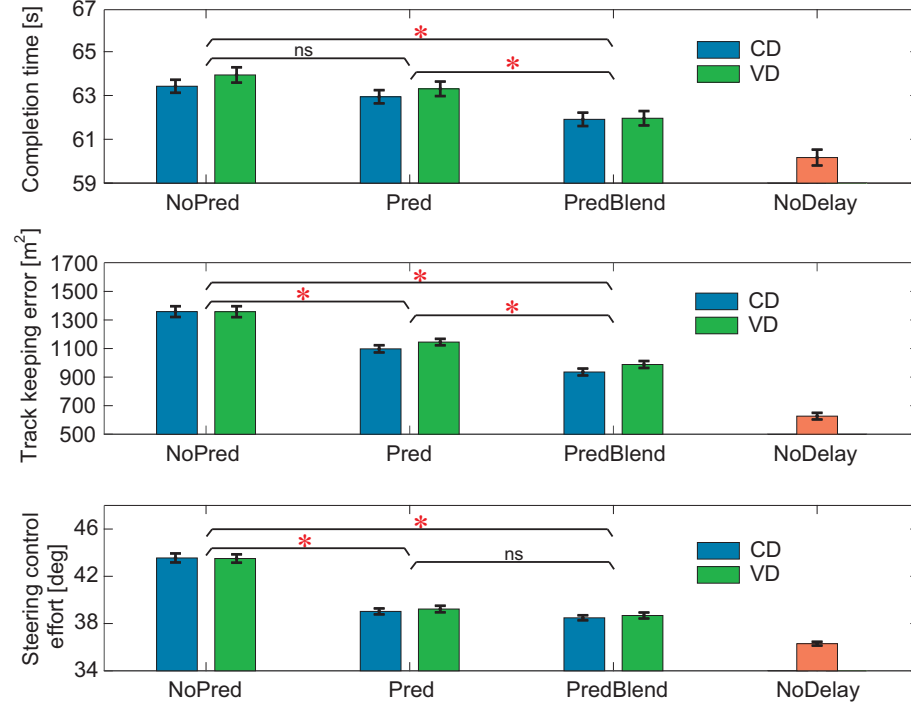


Figure 5.6: Comparison of three performance metrics with different delay types and prediction methods.

world: 18 subjects reported to own driver licenses and 15 of them had more than 2 years of experience to drive the vehicles. However, only 6 subjects were familiar with driving in the virtual environment such as playing racing games. In terms of the situation most similar to this experiment, where human drove the vehicle in simulated environment using steering wheel and pedals, only 3 of them has prior experience. Given the general lack of experience among the test subjects on driving in the used test setup, the importance of training session was emphasized again to help them adapt to the test setup and get used to scenarios especially when large round trip delays of around 0.9 s were added into the system.

The experimental results based on 19 subjects' data are shown in Figure 5.6. Each bar shows the mean and the standard error of mean of all 19 subjects' data in terms of the three performance metrics (*Time, Error, Effort*) under each of the seven scenarios. Smaller metrics indicate better performance. Bar of **NoDelay** is considered as the

Table 5.3: Mean values of metrics under different scenarios with delays.

No Delay	Delay type	CD			VD		
	Prediction method	NoPred	Pred	Pred-Blend	NoPred	Pred	Pred-Blend
60.1	<i>Time</i> [s]	63.4	62.9	61.9	63.9	63.3	62.0
627	<i>Error</i> [m ²]	1357	1098	936	1358	1145	988
1.81	<i>Effort</i> [deg·m]	2.18	1.95	1.92	2.18	1.96	1.93

baseline of performance metrics. Bars using the same prediction method are grouped along the horizontal axis. **NoPred**, **Pred** and **PredBlend** represent the scenario with delays, but without any prediction, with predictors only to perform prediction, and with predictors and blended architecture applied, respectively. Delay type **CD** and **VD** stands for the constant and varying bilateral delays. The mean values of metrics under different scenarios are reported in Table 5.3. It is observed that metrics in scenario **NoPred** without any prediction are much worse than the baseline **NoDelay** and applying prediction methods of **Pred** or **PredBlend** helps reduce all three metrics compared to **NoPred**. The following RM-ANOVA results investigate further whether the difference in means among different scenarios are statistically significant.

RM-ANOVA indicated a significant difference in all three metrics among different prediction methods. F-value and p-value for metrics of *Time*, *Error*, *Effort* are (F=15.54, $p \approx 0$), (F=91.51, $p \approx 0$), and (F=169.69, $p \approx 0$) respectively. Pairwise comparison results are shown in Figure 5.6, where the asterisk means that there exists significant difference in mean pairwise, while “ns” means that the difference in mean is not significant enough compared to the inherent variation in the data.

In terms of the effect of delay type, F-value and p-value for metrics of *Time*, *Error* and *Effort* in the RM-ANOVA results are (F=1.40, $p=0.237$), (F=1.90, $p=0.169$) and (F=0.23, $p=0.628$), respectively. All p-values are greater than 0.05 and hypothesis of existence of no significant difference between constant and varying delays tested

Table 5.4: The level of improvement in performance metrics with **Pred** or **PredBlend**, compared to **NoPred**. "ns" means that no statistically significant improvement is observed in this experiment.

Delay type	CD		VD	
Prediction method	Pred	PredBlend	Pred	PredBlend
Track completion time	15% (ns)	46%	17% (ns)	52%
Track keeping error	36%	58%	29%	51%
Steering control effort	62%	70%	59%	67%

cannot be rejected with 95% confidence level.

Define the level of improvement LoI as:

$$\text{LoI} = \frac{|r_p - r_d|}{|r_{nd} - r_d|} \quad (5.3)$$

where r is the mean of calculated metrics based on all 19 subjects' data in different scenarios, with subscripts p, d, nd representing the scenarios using prediction method of **Pred** or **PredBlend**, without prediction **NoPred**, and no delays **NoDelay**, respectively. The levels of improvement for three metrics are shown in Table 5.4. Compared to **NoPred**, metrics of *Error* and *Effort* are improved significantly by 36% and 62%, respectively with constant delays in scenario **Pred**. With **PredBlend**, metrics of *Time*, *Error* and *Effort* are all improved significantly by 46%, 58% and 70%, respectively. Comparing the two prediction methods developed in this work, vehicle mobility in terms of track completion time and track keeping error are improved more using **PredBlend** than **Pred** by 31% and 22%, respectively, and drivability in terms of steering control effort is improved more by 8% in constant delays. The level of improvement with varying delays are similar.

5.5 Discussion

In this experiment, a round trip delay of around 0.9 s is tested. Note that delays of this magnitude are very large and significantly degrade the closed-loop driving performance even when the test subjects practiced and trained for 1 hour. The metrics are slightly larger under varying delays than constant delays, but the difference is not significant. One possible reason is the delay distribution of the varying delays sequences used in the experiment (Figure 5.4). Note that the sequences are generated based on the measurements of an actual network and are therefore realistic to represent actual network connection, e.g. capturing the heavy tail property in delay distribution. Frequent sudden spikes in the sequences are corresponding to large delay values that do not last for a long duration, thus it is unknown how much effect these spikes would cause on the system. Therefore, with small and fast variations around the mean value, varying delay sequences used in this experiment have an effect similar constant delays in terms of influencing the closed-loop performance. Also, the ratio between standard deviation (σ) and mean (μ) of delays, denoted as $\frac{\sigma}{\mu}$, are less than 0.1, which is relatively small compared to the ratio in other teleoperated driving experiments [18, 59] reporting detectable performance difference between constant and varying delays. For example, in [77, 59], simulated varying delays were generated as a minimum delay plus a one-side zero mean normal distribution and $\frac{\sigma}{\mu}$ used for their experiment were from 0.3 to 0.67. In [18], delays were generated by summing multiple sine waves and vary between 0.4 s and 1.1 s with a mean of 0.7 ms. Such delays were distributed more evenly in the range and potentially had larger $\frac{\sigma}{\mu}$ than 0.1.

Implementing predictor based framework that is developed for delay compensation helps improve the vehicle mobility and drivability significantly compared to the case without prediction (i.e., when framework is not activated). Note that using two model-free predictors only in scenario **Pred**, metrics of *Error* and *Effort* are already improved significantly. This prediction does not rely on any dynamic equations or

parameters about the vehicle or human operator and therefore provides robust performance. In scenario **PredBlend**, a blended architecture is activated in the predictor based framework to help improve the prediction accuracy of vehicle heading, and all three performance metrics are improved even more than those when using predictors only. However, minimal information about the vehicle lateral response is leveraged to implement the model-based *feedforward branch* in the blended architecture, which makes the predictor based framework no longer model-free. Modeling error may affect the performance and robustness may be sacrificed using **PredBlend**.

CHAPTER VI

Field Demonstration and Testing

In this chapter, the developed predictor based framework is implemented on a Mini Baja vehicle platform and preliminary evaluation results are shown.

6.1 Vehicle Platform and Driver Station

Figure 6.1 shows the vehicle teleoperation system used to implement and test the predictor based framework. It is composed of a Driver Station and a Mini Baja vehicle platform that are geographically separated and communicate through radios.

Driver Station contains an operating steering wheel for the human operators to generate steering commands, two Arduino MEGA 2560 microcontrollers, an HP® Z220 Workstation computer (Intel® processor i7-3770, 3.40 GHz) with a monitor to display a driving interface.

The Vehicle Platform is a Mini Baja vehicle originally designed by the student members in the 2008 Michigan Baja Racing team. It was modified by Prof. Brent Gillespie's group to allow the teleoperated steering control capability [129]. It has a 305cc Griggs Stration engine and the maximum speed is around 10 m/s. A modified Logitech® MOMO® steering wheel with an encoder and a motor controller is mounted on the vehicle and acts as the following steering wheel allowing remote control. A high-performance GPS-aided Inertial Navigation System (INS) (VectorNav VN-300)

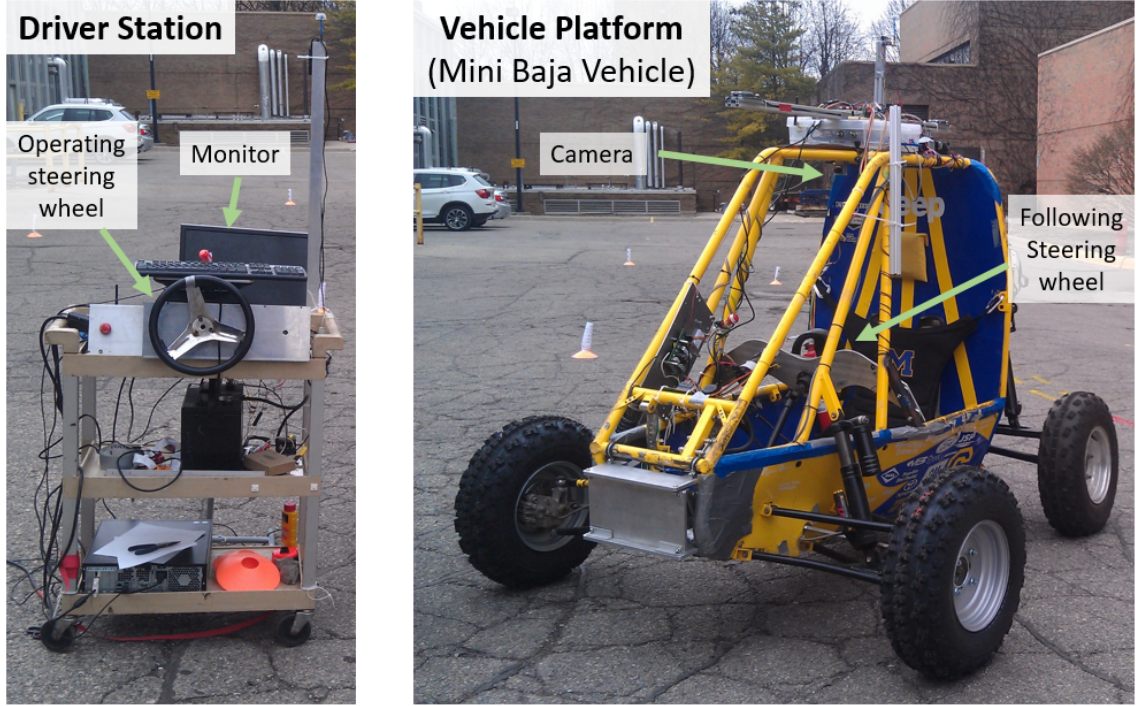


Figure 6.1: Vehicle teleoperation system is composed of a driver station and a Mini Baja vehicle platform with teleoperated steering control capability.

[130] can provide the instantaneous vehicle states up to 400 Hz. Two Arduino MEGA 2560 microcontrollers and one Raspberry Pi 3 Model B+ microcomputer are used as the computing and processing units. A Raspberry Pi Camera Module with Fisheye Lens [131] is also mounted on the vehicle to provide a first-person view for teleoperated driving.

There exist two different communication links between Vehicle Platform and Driver Station. The first link is referred to as the data link and contains a pair of 2.4 GHz nRF24L01 transmitter and receiver modules. It is used to transmit signals such as steering commands and packet send time, which do not require large communication bandwidth, from Driver Station to Vehicle Platform. The second link is referred to as the video link and transmits video including the camera view and vehicle states at high frequency of around 5.8 GHz. A Boscam TS351 transmitter and RC805 receiver are placed at the Vehicle Platform and Driver Station, respec-

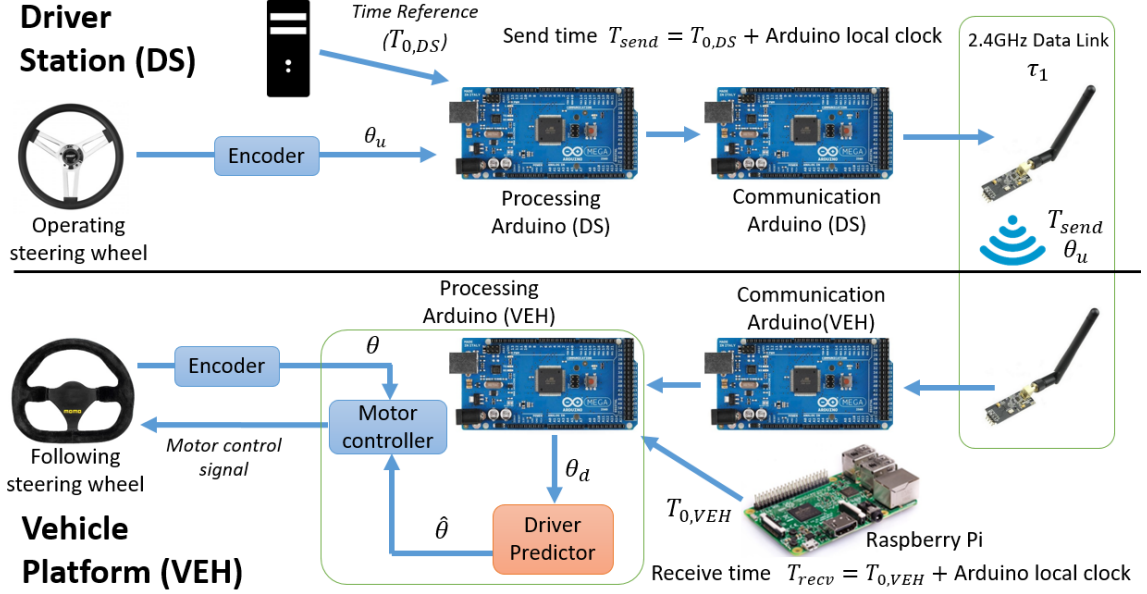


Figure 6.2: One-way prediction of steering commands from Driver Station to Vehicle Platform.

tively, with clover-leaf antennas to enhance the intensity of radio communication and reduce interference from surrounding environment. Considering delays in bilateral communications, bilateral predictions are implemented respectively when applying the predictor based framework to compensate delays and are explained in detail in the following sections.

6.2 Steering Command Prediction

The diagram of predicting vehicle steering is shown in Figure 6.2. At the Driver Station (i.e. transmitter side in this prediction), a steering wheel is operated by a human and its steering angle is measured by an encoder (US Digital optical encoder E3-2048-984-IE-H-D-B) as the undelayed steering command θ_u . An Arduino board is used for processing to receive θ_u and the system time of the computer as Driver Station's absolute time reference $T_{0,DS}$, and pass them to another Arduino board for communication. The send time is determined as an addition of $T_{0,DS}$ to the Arduino

local clock starting when $T_{0,DS}$ is first received. The packets including θ_u and T_{send} are transmitted at a rate of around 100 Hz.

At the Vehicle Platform (i.e. receiver side), from k th received packet, one-way control delay $\tau_1[k]$ is first measured based on the difference between send and receive time. The receive time T_{recv} is calculated in the same way as T_{send} except that the Vehicle Platform time reference is the GPS clock information parsed in Raspberry Pi. Note that synchronization of the system time between Driver Station side and Vehicle Platform side are required for an accurate delay calculation. To achieve that, a fixed time difference between these two sides is measured priorly to convert between absolute time of both sides, assuming the system time does not drift significantly within the couple of minutes that last for a single test run.

A model-free Driver Predictor developed in Chapter III is implemented in a discrete way in an Arduino for processing at the Vehicle Platform side, relying on estimated one-way delay $\tau_1[k]$, delayed steering command $\theta_d[k]$ and delayed steering rate $\dot{\theta}_d[k]$ to generate corresponding predicted steering command $\hat{\theta}[k]$. The derivative is estimated based on numerical differentiation:

$$\dot{\theta}_d[k] = \frac{\theta_d[k] - \theta_d[k-1]}{T_{\text{send}}[k] - T_{\text{send}}[k-1]} \quad (6.1)$$

where $T_{\text{send}}[k]$ is the send time of the k th packet and the same notation holds for $k-1$, $k+1$, etc, as well. Both $\theta_d[k]$ and $\dot{\theta}_d[k]$ are stored in an input buffer.

The predictor dynamics implemented in a digital system like Arduino is in the form of a discrete difference equation:

$$\theta_p[k+1] = \theta_p[k] + (T_{\text{recv}}[k+1] - T_{\text{recv}}[k])(\dot{\theta}_d[k] + \lambda(\theta_d[k] - \hat{\theta}_{\text{interp}}[k])) \quad (6.2)$$

where $\cdot[k]$ refers to the corresponding signal for k th arrived packet, θ_d and $\dot{\theta}_d$ are the delayed predictor inputs, θ_p is the predictor state, $\hat{\theta}$ is the predictor output, and $\hat{\theta}_d$

is the retarded predictor output delayed by measured τ_1 . Note that the receive time T_{recv} , θ_p and $\hat{\theta}$ are stored in a buffer so that $\hat{\theta}_{\text{interp}}[k]$ can be obtained based on the interpolation between two past predictor outputs $\hat{\theta}[m]$ and $\hat{\theta}[n]$ in the buffer, where $m, n \leq k$ and the corresponding send time $T_{\text{recv}}[m]$ and $T_{\text{recv}}[n]$ satisfy:

$$T_{\text{recv}}[m] < T_{\text{recv}}[k] - \tau_1[k] \leq T_{\text{recv}}[n] \quad (6.3)$$

Similarly, the predictor output equation is also expressed in discrete form. In the original predictor form, $\hat{\theta}[k] = \theta_p[k]$. Applying saturation and resetting scheme as in Section 3.6, the output equation becomes:

$$\hat{\theta}[k] = f_{\text{sat}}(\theta_p[k], \dot{\theta}_d[k]) \quad (6.4)$$

where $f_{\text{sat}}(\theta_p[k], \dot{\theta}_d[k])$ deals with saturation according to

$$f_{\text{sat}}(\theta_p[k], \dot{\theta}_d[k]) = \begin{cases} \min(\theta_p[k], \hat{\theta}_{\text{sat}}[k]) & \text{if } \dot{\theta}_d[k] \geq 0 \\ \max(\theta_p[k], \hat{\theta}_{\text{sat}}[k]) & \text{if } \dot{\theta}_d[k] < 0 \end{cases} \quad (6.5)$$

and $\hat{\theta}_{\text{sat}}[k] = \frac{\dot{\theta}_d[k]}{\lambda} + \theta_d[k]$. Also, the predictor state $\theta_p[k]$ is reset to be the same as the delayed steering command $\theta_d[k]$ when

$$\begin{cases} \dot{\theta}_d[k] \text{ changes from } (+) \text{ to } (-) \text{ and } \theta_p[k] \geq \hat{\theta}_{\text{sat}}[k] \\ \dot{\theta}_d[k] \text{ changes from } (-) \text{ to } (+) \text{ and } \theta_p[k] < \hat{\theta}_{\text{sat}}[k] \end{cases} \quad (6.6)$$

The predictor parameter λ is selected based on the design procedure developed in Section 3.7.

Finally, a proportional controller is designed in the Processing Arduino, aiming to generate a Pulse Width Modulation (PWM) control signal V_{PWM} that drives the fol-

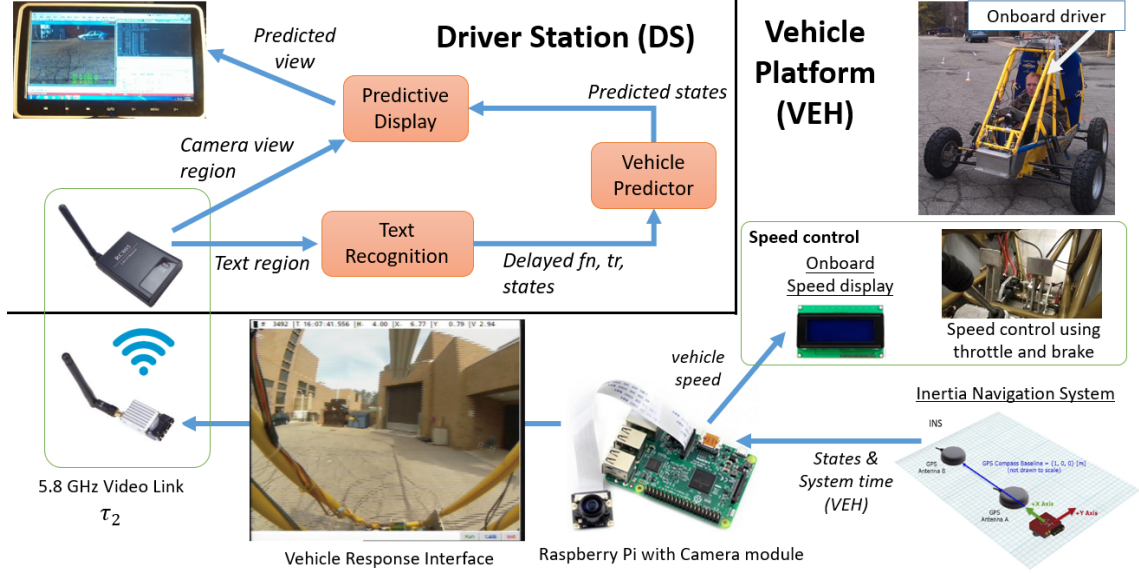


Figure 6.3: One-way prediction of vehicle states and camera view from Vehicle Platform to Driver Station.

lowing steering wheel on the vehicle to reach the same rotary position as the operating steering wheel controlled by human operators at the Driver Station. The control law is shown below:

$$V_{\text{PWM}} = k_s(\hat{\theta} - \theta) \quad (6.7)$$

where θ is the current steering wheel angle, $\hat{\theta}$ is the predicted reference angle to follow using Driver Predictor, and proportional gain $k_s = \frac{0.1 * 20000}{\pi}$. V_{PWM} is saturated between -230 and 230 to control the DC motor voltage that drives the following steering wheel.

6.3 Vehicle State and Camera View Prediction

The diagram of predicting vehicle states and generating predicted camera view is shown in Figure 6.3. At the Vehicle Platform (i.e. transmitter side in this prediction), vehicle states of heading, global position and longitudinal speed are measured through the VectorNav sensor. Two GPS receivers with antennas are placed along

the centerline of the vehicle and separated by 1.12 m, providing individual position information at 5Hz in the coordinates of both Earth-Centered, Earth-Fixed (ECEF) and Latitude, Longitude, Altitude (LLA) coordinate frames. One of the benefits of a dual GPS is that vehicle heading can be estimated very accurately and the estimation is more accurate as the distance between the two GPS receivers increases. An Extended Kalman Filter is embedded in the INS, estimating all the abovementioned vehicle states based on combined measurement from the dual GPS and IMU sensors. Note that the states of global positions are converted into the coordinates of East-North-Up (ENU) (i.e. x in east direction and y in north direction) based on measurements X, Y, Z in ECEF coordinates, latitude ϕ and longitude γ in the following way [132]:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} -\sin \gamma_r & \cos \gamma_r & 0 \\ -\sin \phi_r \cos \gamma_r & -\sin \phi_r \cos \gamma_r \cos \phi_r & \cos \phi_r \end{bmatrix} \begin{bmatrix} X - X_r \\ Y - Y_r \\ Z - Z_r \end{bmatrix} \quad (6.8)$$

where subscript r is related to the reference position. The reference position is determined when the following two conditions are met for the first time:

1. The status of the GPS receivers indicates that they detect and track more than 5 satellites simultaneously.
2. Vehicle speed is greater than some negligible value like 0.5 m/s, so that vehicle states are estimated in a moving vehicle base mode.

To avoid the need for synchronizing the camera view and the corresponding vehicle states when each frame of camera view is captured, up-to-date vehicle states are displayed alongside the camera view in the same video frame for transmission. A vehicle response interface is thus implemented and displayed in full screen in the Raspberry Pi with a Python script to overlay the state information onto the camera

view. An example frame of the interface is shown in Figure 6.4 and is divided into a text region and a camera view region.

The camera view region contains a first person view with resolution of 700x525 pixels (aspect ratio 4:3) captured from a Raspberry Pi Camera Module with a horizontal field of view adjusted to be 126 degrees. Using Python Picamera Library, the camera view is previewed in 25 Hz.

The starting horizontal and vertical edge of the text region is determined by identifying the first character filled with all black. Each character is displayed in a monospaced Courier New font and occupies a box of 10x12 pixels. From left to right, information of vehicle response shown in the frame are the frame number fn , GPS time tr , and all the vehicle states of heading ψ , global X and Y position x, y and vehicle longitudinal speed u . Their position and width are formatted in a deterministic way for the purpose of text recognition. Texts are updated in 25 Hz, as well, to synchronize the associated vehicle states with the camera view. The vehicle response interface is encoded in PAL at 25 Hz and transmitted to the Driver Station with a 5.8 GHz video link.

Note that speed u is also displayed to an on-board driver on a LCD. There are two reasons to include the on-board driver. First, the vehicle currently does not have the functionality of changing speed in the way of drive-by-wire. Thus, the on-board driver is responsible for pressing throttle and brake pedals to accelerate/decelerate the vehicle and maintain constant vehicle speed in the experiments shown in Section 6.4, but vehicle steering is still controlled remotely by the human operator at the Driver Station. The second reason is safety, as the on-board driver can intervene and perform an emergency maneuver if needed.



Figure 6.4: The vehicle response interface is composed of a text region with vehicle states and system time, and a camera view region.



	fn	tr
Clear texts:	7469	14:45:15.629
Unclear texts:	747 	14:45:15. 

Figure 6.5: Difference between clear texts and unclear texts. Unclear text regions are marked with a rectangular box.

6.3.1 Vehicle State Recognition

The transmitted vehicle response interface is captured in the computer at the Driver Station, using a USB video capture adapter [133]. Vehicle states in the text region are recognized in Visual Studio 2015. The ground truth pixel values in gray scale from 0-255 of all the necessary characters (i.e. number 0 to 9, ‘+’ and ‘-’) are saved a priori. Therefore, recognizing each character in the text region can be achieved by comparing it to all the saved ground truth characters and counting *Simpixel*, the total number of pixels with similar colors (i.e. difference of pixel values less than a threshold). The character is thus detected as a certain number or sign with maximum *Simpixel*.

After recognition, texts are considered to be clear when each character in the texts can be detected and mapped to one single ground truth character. However, when the transmitted vehicle response interface is captured between two frames, the text region is a combination of adjacent frames and the text sometimes is considered to be unclear. An example of clear and unclear texts including frame number fn and GPS time tr are shown in Figure 6.5. tr is shown in the format from hour to minute and to second with resolution of 1 ms. In the second line classified as unclear texts, fn seems to be between 7470 and 7471, while tr is likely to transition from 14:45:15.669 to 14:45:15.709. In such case, two most likely characters are detected and the exact detection is determined based on the coupling between fn and tr . Since fn and tr are updated in 25 Hz in each frame, increasing fn by 1 corresponds to an increase in

tr by 40 ms. Thus, fn and tr form specific combinations, i.e. $\{7469, 14:45:15.629\}$, $\{7470, 14:45:15.669\}$ and $\{7471, 14:45:15.709\}$. The combinations with smallest fn but greater than the previous recognized frame number are determined so that always the latest packets are captured and used for prediction. Vehicle states ψ, x, y, u are recognized in the same way as introduced above and these delayed states are stored in a predictor input buffer.

6.3.2 Vehicle State Prediction and Predicted Camera View

Considering the robustness of text recognition, any sudden change in fn , tr and vehicle states are corrected (by recognizing with different thresholds) or considered as dropped. Also, the vehicle states are filtered by a low-pass filter with cutoff frequency of 1 Hz to attenuate the high frequency noise when performing estimation of the derivatives of the vehicle states. Based on delayed states and state derivatives, each vehicle state is predicted in the Vehicle Predictor individually based on the same form as the Driver Predictor in (6.2), with design options of selecting λ and whether to include saturation and resetting scheme as in (6.5) and (6.6). The benefit of model-free predictors are highlighted when dealing with a physical vehicle instead of a known simulated vehicle model: no information about the vehicle dynamics, powertrain or tire characteristics are required for prediction of vehicle states.

What human operators rely on to drive the vehicle through teleoperation is the camera view. However, camera views included in the frames of the vehicle response interface are delayed. A predicted display algorithm developed in [22] is leveraged in this work. The algorithm performs perspective transformation to predict what a camera would see from a predicted location given the delayed camera view. Predicted states are converted into the local coordinates specified by the delayed states and the difference in positions and heading direction, dx, dy, dh are determined. Thus, if the vehicle states predicted by the Vehicle Predictor are close to the undelayed states,

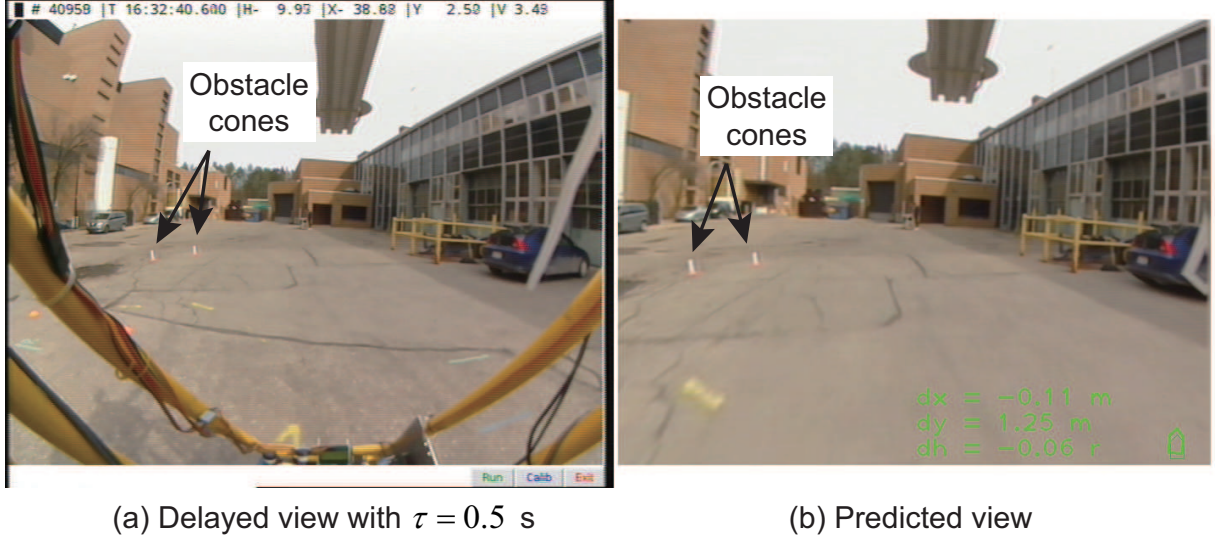


Figure 6.6: Comparison between delayed and predicted camera view.

the predicted camera view displayed to the human operators is similar to the camera view transmitted without delays. An example comparison between (a) delayed camera view and (b) predicted camera view with dx, dy, dh displayed on the view is shown in Figure 6.6. With $dy = 1.25$ m > 0 in the example frame, predicted camera view captures the environment in a position 1.25 m farther than seen from the delayed view. As an example, the two obstacle cones look bigger and closer to the vehicle, and this distance to the vehicle is closer to that in the actual environment, when there is no delay in the camera view.

6.4 Preliminary Results of Closed-Loop Experiments

In this section, preliminary results of closed-loop teleoperation experiments are presented to mainly show the potential of the model-free predictors implemented on actual hardware and vehicle platform to compensate delays and improve vehicle mobility and drivability.

In this preliminary closed-loop experiment, only the one-way communication in the data link when the steering commands are transmitted from Driver Station to

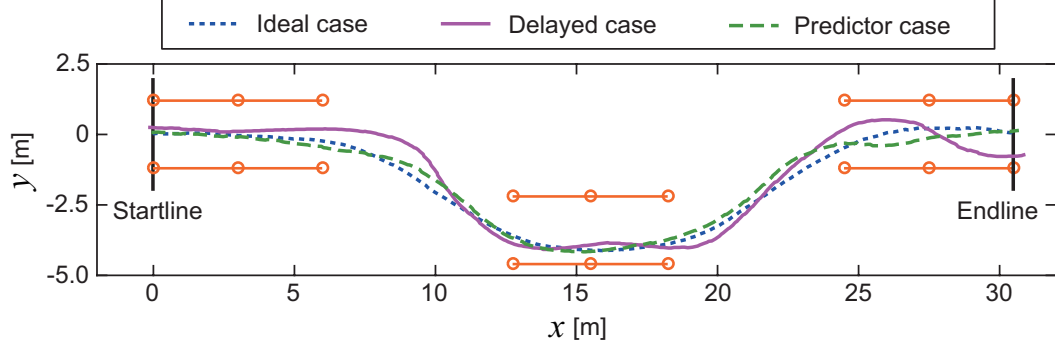


Figure 6.7: Comparison among vehicle trajectories of ideal, delayed and predicted cases in a double lane change scenario.

Vehicle Platform was used, and a human operator controlled the vehicle in direct line of sight.

A double lane change scenario (with half of the total maneuver length compared to the standard one [134]) has been tested. Locations of the obstacle cones and the start and end line are shown in Figure 6.7. Three runs are performed by one human driver on three test cases: ideal case without any delays, delayed case with varying delays (mean around 0.5 s), and predicted case with the same amount of delays but with a Driver Predictor ($\lambda = 0.4\lambda_{\max}(\tau_{\text{avg}} = 0.5)$) with saturation and resetting scheme). The vehicle speed is maintained at around 2 m/s by an on-board driver. Vehicle mobility in the aspect of vehicle's lateral performance, including the vehicle trajectory and steering control effort, are studied.

The vehicle trajectories of these three cases are also shown in Figure 6.7. The trajectory of the ideal case is very similar to a standard double lane change maneuver and is considered as the baseline of teleoperated driving performance. The trajectory in the delayed case shows deviations from the baseline, as well as frequent changes in the vehicle heading. These frequent changes occur, because with delays there exists asynchrony between the human sending steering commands and noticing the corresponding change in the angle of vehicle tires, causing the human operator to oversteer and then compensate the steering by themselves. Compared to the delayed

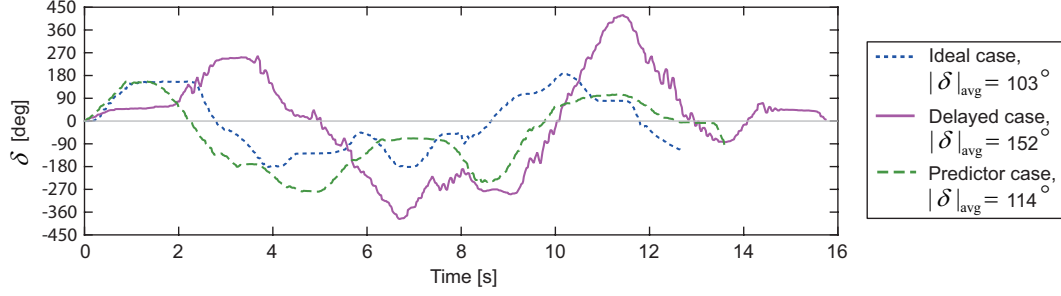


Figure 6.8: Comparison among vehicle steering wheel angle δ of ideal, delayed and predicted cases in a double lane change scenario.

case, the trajectory in the predictor case is closer to that in the ideal case and also has fewer changes in the vehicle heading.

Figure 6.8 shows the time history of the angles of the following steering wheel δ on the vehicle among the three cases. Note that there exists small oscillations in the delayed case due to packets arriving with varying delays, and the predictor helps to reduce these oscillations as reported in Section 3.4.2.3. Steering control effort $|\delta_{\text{avg}}|$ is calculated as the average magnitude of the angle of the following steering wheel on the vehicle. $|\delta_{\text{avg}}|$ for ideal, delayed and predicted cases are 103 deg, 152 deg and 114 deg, respectively. The Driver Predictor helps with reducing steering control effort under delays. Note that in Figure 6.8, the length of the steering angle profiles represents the completion time in each of three cases. It seems that the completion time in the predictor case is less than that in the delayed case, however there is no direct evidence indicating that vehicle mobility in terms of longitudinal performance of completion time is improved with the predictor, because in the scenario, an on-board driver instead of the human operator at the Driver Station is maintaining the constant speed, which may vary from case to case.

Thus, preliminary results tested in the field show that adding a model-free Driver Predictor to predict steering commands appears to improve the mobility and drivability in the aspect of lateral performance of the vehicle, i.e. a smoother vehicle trajectory and less steering control effort compared to without prediction.

6.5 Current Issues and Future Work

When conducting the tests, issues due to the existing hardware and algorithm are identified as below. Some potential ideas as the future work are also provided to address these issues.

6.5.1 Video Transmission Quality

Currently, there exists a limitation on quality of video transmission and capturing. Note that the 5.8 GHz video link is commonly used on Unmanned Aerial Vehicles (UAVs) to send real time first person view on UAVs back to the ground station and can be applied to transmit camera view of the ground vehicles in this work, as well. However, the limitation is that there exists much more interference in the environment on the ground than up in the air with clear line-of-sight, due to more objects as the obstacles. As an example, the video link used in this work with 200 mW transmission power can transmit steady and clear image frames up to several kilometers with clear line-of-sight for UAV applications, but when tested with the Mini Baja Vehicle, the image frames start to be distorted and may even not be received successfully (drop during communication) when the transmission distance is larger than 20 meters. This is because radio signals transmitted in the high frequency of 5.8 GHz have weak penetration through objects. Also, sometimes the frames are blurred or blocked by the vertical scanning line when using the video capture adapter to capture the frame in PAL format. There may exist around 5% false recognition rate in the current algorithm when dealing with these blurry frames. The distorted or blurry frames therefore add difficulty in recognizing the text regions correctly for state prediction.

Potential improvements on video transmission quality include:

1. Using multiple antennas to receive video frames. Unidirectional antennas can

be used along with existing omnidirectional antennas on the vehicle to provide extended range and more consistent signal transmission quality. The energy of radio wave is concentrated using unidirectional antennas and high signal strength and thus good transmission quality can be achieved in the specified direction where antennas point toward. Omnidirectional antennas such linear whip and circular polarized (CP) antennas transmit and receive signal evenly in all directions. Linear whip antennas transmit radio wave in one plane and best reception is achieved when the antennas of both the transmitter and the receiver align in the same plane. CP antennas, on the contrary, transmit the radio wave with circular planes, formed by a combination of two orthogonal planes. Therefore, signals can be received regardless of the directions that antennas point towards. Also, multi-path signals (due to bouncing between surfaces in the environment) could be rejected. But their gains are reduced by 3 dB due to energy split into two directions, and this leads to smaller range compared to linear antennas. Thus, the received video quality can be improved when the 5.8 GHz receiver is attached to both an unidirectional antenna and a CP antenna.

2. Using video link with lower transmission frequency. Video link with lower transmission frequency such as 1.2 GHz is also feasible to transmit analog video frames. Transmitting in 1.2 GHz has the benefit of enhancing the penetration capability through objects, which seems critical to the actual close-to-ground environment where the vehicle is moving in with many objects as obstacles. On the other hand, potential drawbacks include sacrificing the transmission bandwidth and interference by the frequency band of 2.4 GHz (which is the resonate frequency of 1.2 GHz).
3. Testing in a completely open space, where there is not many objects like build-

ings or trees as obstacles

Also, an alternative solution can be to transmit the information in the text region in a separate channel in addition to being included in the video frame, which would avoid the issues associated with recognizing all vehicle states from time to time from inconsistent video frames. Vehicle states can be transmitted in bytes at a faster and reliable way and saved in a buffer. Recognition would only be done on either the frame number or GPS time to pair with the history values stored in the buffer and the vehicle states synchronized to the received camera view would thus be directly available. This way prediction performance would be less affected by the unreliable video link.

6.5.2 Vehicle Speed Control

Currently, the speed control of the Mini Baja vehicle is not drive-by-wire. An on-board driver is necessary to accelerate or decelerate the vehicle with throttle and brake pedals. To make the vehicle have full teleoperation capability, i.e. with speed and steering remotely controlled, additional actuators and controllers need to be implemented so that human operators at the Driver Station can provide either direct commands of throttle and brake, or a reference speed for the vehicle speed controller to follow.

6.5.3 Steering Haptic Feedback

No haptic force feedback has been implemented yet on the operating steering wheel. Without it, human operators cannot feel the vehicle response when the steering wheel is mechanically linked to the tires that contact the ground. This increases the difficulty in teleoperated driving. Haptic feedback can be achieved by transmitting the following steering wheel angle on the vehicle back to the Driver Station using the 2.4 GHz data link as in [129] so that a virtual haptic torque can be added to the

operating steering wheel to provide some haptic response to help teleoperated driving. However, due to additions of send time information required by the predictors, more time are spent on transmitting signals of steering angles and send time between the Arduino for processing and the Arduino for communication, as well as transmitting them through the data link. The motor control is operated up to around 64 Hz, which is relatively slower than a suggested frequency of over 100 Hz when the current motor setup is capable of providing a smooth haptic torque without causing noticeable jerk in the steering wheel. If vehicle states are included in the transmission as proposed in Section 6.5.1, the frequency would be even lower.

In addition, it remains an open research question to generate such haptic torque when significant communication delays in the data link (such as an additional virtual network delay of 0.5 s in Section 6.4) are considered.

CHAPTER VII

Conclusions and Future Work

7.1 Dissertation Summary

Teleoperated Unmanned Ground Vehicles (UGVs) have been widely used in applications when driver safety, mission efficiency or mission cost is a major concern. One major challenge with teleoperating a UGV is that communication delays can significantly affect the mobility performance of the vehicle and make teleoperated driving tasks very challenging especially at high speeds.

In this dissertation, a predictor based framework with predictors of a new form and a blended prediction architecture is developed to compensate effects of delays through signal prediction, thereby improving vehicle mobility performance. The novelty of the framework is that minimal information about the governing equations of the system is required in prediction to benefit from performance robustness to modeling errors.

This dissertation first investigates a model-free prediction solution and develops a predictor that does not require information about the vehicle dynamics or human operators' motion for prediction, as presented in Chapter III. Compared to the existing model-free methods, neither assumptions about the particular way the vehicle moves, nor knowledge about the noise characteristics that drive the existing predictive filters are needed. Its stability and performance are studied and a predictor design procedure is presented. A saturation and resetting scheme is developed to further improve

the predictor transient performance. Also, predictor can potentially help stabilize the unstable closed-loop system.

Secondly, in Chapter IV, a blended prediction architecture is developed to blend the outputs of the model-free predictor with those of a steering feedforward loop that relies on minimal information about vehicle lateral response. Better prediction performance is observed based on open-loop tests with the blended architecture compared to using either the model-free predictors or the model-based feedforward loop alone.

The mobility performance of teleoperated vehicles with delays and the predictor based framework are evaluated in this dissertation in Chapter V and Chapter VI with human-in-the-loop experiments using simulated and physical vehicles, respectively in teleoperation mode. Predictor based framework is proven to be beneficial in statistically significantly improving vehicle mobility and drivability in the experiments performed.

7.2 Conclusions and Original Contributions

Based on the simulation based experiments under a track following scenario, the predictor-based framework with two model-free predictors implemented on high-speed teleoperated UGVs with large round trip delays of 0.9 s can improve the mobility metric of track keeping error by 33% and drivability metric of steering control effort by 61% compared to without predictors to compensate delays. Minimal information about the vehicle lateral response is included in the developed blended prediction architecture to improve prediction accuracy of vehicle heading, at the potential cost of robustness due to the addition of system information. The framework with predictors and the blended architecture results in a larger improvement in both mobility and drivability of the vehicle: track completion time, track keeping error and steering control effort are improved by 48%, 55%, and 69% compared to no prediction. Preliminary results of the field test under a double lane change maneuver show that

predictors appear to improve the vehicle lateral performance as well as the drivability under delays.

This dissertation makes the following original contributions to the literature.

1. A novel model-free predictor is developed to predict signals and compensate communication delays (Chapter III). It is completely model-free in the sense that no information about the governing equations of any component in the system is used in the prediction. Also, in contrast to other model-free prediction methods, no assumptions about the system dynamics or knowledge about the noise characteristics that drive the existing predictive filters are required. The development of this framework entails the following contributions.

With constant delays,

- 1.1. predictor performance in steady state [82] and transient [83] is analyzed.
A saturation and resetting scheme is developed to improve transient performance with modifications on the predictor dynamics [111, 84].
- 1.2. a design procedure to select predictor parameters is developed [84].
- 1.3. closed-loop stability is studied based on LTI system [83].

With varying delays,

- 1.1. predictor stability with varying delays is established [83].
- 1.2. performance robustness of the predictor to varying delays and network effects are evaluated and initial results for closed-loop stability are provided based on a general networked system [83].
2. A predictor based framework is developed for teleoperated vehicles to improve their mobility performance under large delays. (Chapter IV, V, VI)
 - 2.1. A blended prediction architecture is developed to blend the predictions from a model-based feedforward branch with the predictions from a model-

free feedback branch for better prediction accuracy of vehicle heading and robustness to modeling error [84].

- 2.2. The predictor based framework is evaluated based on human-in-the-loop experiments in a simulated environment [111, 115] and real world using a teleoperated vehicle.

7.3 List of Publications

The following list provides the papers published associated to these contributions and work present in this dissertation:

1. [83] **Y. Zheng**, M.J. Brudnak, P. Jayakumar, J.L. Stein, and T. Ersal. A Predictor Based Framework for Delay Compensation In Networked Closed-Loop Systems. IEEE/ASME Trans. Mechatronics (In review), 2018.
2. [84] **Y. Zheng**, M.J. Brudnak, P. Jayakumar, J.L. Stein, and T. Ersal. A Delay Compensation Framework for Predicting Heading in Teleoperated Ground Vehicles. IEEE/ASME Trans. Mechatronics (In submission), 2018.
3. [115] **Y. Zheng**, M.J. Brudnak, P. Jayakumar, J.L. Stein, and T. Ersal. Human-in-the-loop Experimental evaluation of a Predictor Based Framework for Teleoperated Unmanned Ground Vehicles. In preparation, 2018.
4. [111] **Y. Zheng**, M.J. Brudnak, P. Jayakumar, J.L. Stein, and T. Ersal. An Experimental Evaluation of a Model-Free Predictor Framework in Teleoperated Vehicles. IFAC-PapersOnLine, 49(10):157-164, 2016.
5. [106] X. Ge, **Y. Zheng**, M.J. Brudnak, P. Jayakumar, J.L. Stein, and T. Ersal. Performance analysis of a model-free predictor for delay compensation in networked systems. IFAC-PapersOnLine, 48(12):434439, 2015.

6. [82] X. Ge, **Y. Zheng**, M.J. Brudnak, P. Jayakumar, J.L. Stein, and T. Ersal. Analysis of a Model-Free Predictor for Delay Compensation in Networked Systems. *Time Delay Systems*, pages 201215, 2017.

7.4 Future Work

The following subsections describe several areas of future work.

7.4.1 Compensating Visual Delays for Semi-Autonomous UGVs

For UGVs operated in semi-autonomous mode, communication delays remain as one of the challenges that could affect the vehicle mobility performance. For human operators, while controlling vehicle may be affected less by delays with the assistance of automation, the fidelity of monitoring the vehicle response is reduced with delays. Thus, the methods developed in this work can potentially be applied on semi-autonomous UGVs to compensate visual delays. Specifically, the model-free predictors can be implemented to predict the vehicle states for the purpose of generating predicted camera view. Some of the preliminary results are given below and inspire for future development.

A haptic shared control scheme developed in [129, 135] was used to test the predictors' utility in the semi-autonomous scheme. This haptic shared control scheme allows both human and automation to cooperate on performing control actions (shared control) all the time using motorized steering wheels to establish the haptic communication between human and automation. Haptic shared control scheme without visual delays and with delays of 0.6 s was tested in a simulation based human-in-the-loop experiment using the motorized steering wheel setup from Prof. Brent Gillespie's HaptiX lab and a model predictive control automation algorithm based on [136]. The test scenario was that human and automation cooperated based on this haptic scheme to control a vehicle in constant speed to follow the centerline of the track while avoid-

ing obstacles. The setup for prediction was similar to that in Chapter V: A Vehicle Predictor was added at the Driver Station, predicting the vehicle states of position and heading, and then the camera view associated to the predicted vehicle position in the virtual environment was displayed to the operators.

6 human test drivers' data indicated that applying the Vehicle Predictor with 0.6 s of visual delays helped significantly with reducing the performance metric of average track keeping error by 63% and metric of average steering control effort by 70% [137, 138]. Thus, predictors developed in this work show potential in compensating visual communication delays when human operators monitor the response of a semi-autonomous UGVs. Future research could focus on compensating the delays when the control actions generated by the automation on-board are transmitted to the Driver Station for haptic communication between human and automation.

7.4.2 Theoretical Development of Predictors

The predictor developed in this work is most effective for low-frequency signal inputs. Predictor dynamics can be further studied to increase the predictor bandwidth with modified structure while still preserving the model-free property.

Additionally, it is an open research question whether predictor parameters λ and blending weight α in the blended architecture can be adaptively tuned during closed-loop teleoperated driving. Predictor performance is frequency-dependent and there exist tradeoffs in λ when discussing predictor's steady state and transient performance. Adaptive λ corresponding to frequency of the signal to predict is worth studying. In the blended architecture, α can also be adaptive to maximize the robust prediction accuracy of vehicle heading.

The predictor stability has been addressed and preliminary results have shown the benefit of predictors in terms of stabilizing an LTI closed-loop system. For teleoperated vehicle system with nonlinear vehicle platforms and human operators in the

loop, its closed-loop stability remains to be studied. Wave variable transformation in the telerobotics literature can be leveraged to passify the communication channels regardless of delays. Then the closed-loop teleoperated vehicle system stability can be established with robustness to delays if the passivity of each sub component in the system (i.e. human operators to do teleoperated driving with delays, vehicle platform, predictors with blended architecture) can also be guaranteed. Accordingly, performance loss due to establishment of robust closed-loop stability remains to be investigated.

7.4.3 Field Test With Mini Baja Vehicle

The field test results show that the developed model-free predictor is promising to improve vehicle mobility (lateral performance alone) and drivability of the Mini Baja vehicle operated in teleoperation under delays. However, the field test is not completed due to time and hardware limitations listed in Section 6.5. Future work involves improving the quality of video transmission, potentially implementing the blended architecture to predict vehicle heading with accuracy and robustness, and performing human-in-the-loop closed-loop experiments with bilateral delays and predictions on transmitted control commands and vehicle response.

BIBLIOGRAPHY

BIBLIOGRAPHY

- [1] D. W. Gage. UGV HISTORY 101: A Brief History of Unmanned Ground Vehicle (UGV) Development Efforts. *Unmanned Systems*, 13:9–32, 1995.
- [2] S Lichiardopol. A Survey on Teleoperation. *Eindhoven University of Technology, DCT report*, 2007.
- [3] Defense Department and Dasch, J.M. and Gorish, D.J. and Roddin, M. and Van Enkenvoort, R. and U.S. Army Tank-Automotive Research, Development, and Engineering Center. *The TARDEC Story: Sixty-five Years of Innovation 1946-2010*. 2013.
- [4] U.S. Army Training and Doctrine Command. The U.S. Army Robotic and Autonomous Systems Strategy. 2017.
- [5] J.Y.C. Chen, E.C. Haas, and M.J. Barnes. Human performance issues and user interface design for teleoperated robots. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 37(6):1231–1245, 2007.
- [6] D.W. Cunningham, A. Chatziastros, M. von der Heyde, and H. H. Bülthoff. Driving in the future: temporal visuomotor adaptation and generalization. *Journal of Vision*, 1(2):88–98, 2001.
- [7] K.U. Smith. *Delayed sensory feedback and behavior*. Oxford, England: W. B. Saunders, 1962.
- [8] T.B. Sheridan. Space teleoperation through time delay: Review and prognosis. *IEEE Transactions on Robotics and Automation*, 9(5):592–606, 1993.
- [9] W.R. Ferrell. Remote manipulation with transmission delay. *IEEE Transactions on Human Factors in Electronics*, 6(1):24–32, 1965.
- [10] L.H. Frank, J.G. Casali, and W.W. Wierwille. Effects of visual display and motion system delays on operator performance and uneasiness in a driving simulator. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 30(2):201–217, 1988.
- [11] T.B. Sheridan. *Telerobotics, automation, and human supervisory control*. MIT Press: Cambridge, MA, 1992.

- [12] M. Maurette, L. Boissier, M. Delpech, C. Proy, and C. Quere. Autonomy and remote control experiment for lunar rover missions. *Control Engineering Practice*, 5(6):851–857, 1997.
- [13] N. Murakami, A. Ito, J.D. Will, M. Steffen, K. Inoue, K. Kita, and S. Miyaoura. Development of a teleoperation system for agricultural vehicles. *Computers and Electronics in Agriculture*, 63(1):81–88, 2008.
- [14] G. Witus, S. Hunt, and P. Janicki. Methods for UGV teleoperation with high latency communications. In *Proceedings of the SPIE, Unmanned Systems Technology XIII*, volume 8045N, 2011.
- [15] I. Elhajj, N. Xi, W.K. Fung, Y.H. Liu, W.J. Li, T. Kaga, and T. Fukuda. Haptic information in internet-based teleoperation. *IEEE/ASME Transactions on Mechatronics*, 6(3):295–304, 2001.
- [16] X. Zhu, Z. Li, W. Ye, and H. Zhao. Event-based tele-presence of a mobile service robot. In *2011 IEEE International Workshop on Haptic Audio Visual Environments and Games*, pages 118–123, 2011.
- [17] S. Trimpe and J. Buchli. Event-based estimation and control for remote robot operation with reduced communication. In *IEEE International Conference on Robotics and Automation*, pages 5018–5025, 2015.
- [18] J. Davis, C. Smyth, and K. McDowell. The effects of time lag on driving performance and a possible mitigation. *IEEE Transactions on Robotics*, 26(3):590–593, 2010.
- [19] A. Kelly, N. Chan, H. Herman, D. Huber, R. Meyers, P. Rander, R. Warner, J. Zigar, and E. Capstick. Real-time photorealistic virtualized reality interface for remote mobile robot control. *The International Journal of Robotics Research*, 30(3):384–404, 2011.
- [20] F. Chucholowski, S. Buechner, J. Reicheneder, and M. Lienkamp. Prediction Methods for Teleoperated Road Vehicles. In *2013 Conference on Future Automotive Technology*, 2013.
- [21] F. Chucholowski. Evaluation of Display Methods for Teleoperation of Road Vehicles. *Journal of Unmanned System Technology*, 3(3):80–85, 2016.
- [22] M. Brudnak. Predictive Displays for High Latency Teleoperation. In *2016 Ndia Ground Vehicle Systems Engineering and Technology Symposium*, 2016.
- [23] C.L. Lai and P.L. Hsu. Design the remote control system with the time-delay estimator and the adaptive smith predictor. *IEEE Transactions on Industrial Informatics*, 6(1):73–80, 2010.
- [24] A. Alvarez-Aguirre. Predictor Based Control Strategy for Wheeled Mobile Robots Subject to Transport Delay. *Remote Telerobotics*, pages 33–59, 2010.

- [25] E. Slawiński, V.A. Mut, and J.F. Postigo. Teleoperation of mobile robots with time-varying delay. *IEEE Transactions on Robotics*, 23(5):1071–1082, 2007.
- [26] C. Smith and P. Jensfelt. A predictor for operator input for time-delayed teleoperation. *Mechatronics*, 20(7):778–786, 2010.
- [27] X.R. Li and V.P. Jilkov. Survey of maneuvering target tracking. Part I. Dynamic models. *IEEE Transactions on Aerospace and Electronic Systems*, 39(4):1333–1364, 2003.
- [28] C. Wang, C. Lin, and M. Tomizuka. Statistical learning algorithms to compensate slow visual feedback for industrial robots. *Journal of Dynamic Systems, Measurement, and Control*, 137(3):031011, 2015.
- [29] R.J. Anderson and M.W. Spong. Bilateral control of teleoperators with time delay. *IEEE Transactions on Automatic Control*, 34(5):494–501, 1989.
- [30] G. Niemeyer and J. Slotine. Stable adaptive teleoperation. *IEEE Journal of Oceanic Engineering*, 16(1):152–162, 1991.
- [31] S. Munir and W. J. Book. Internet-based teleoperation using wave variables with prediction. *IEEE/ASME Transactions on Mechatronics*, 7(2):124–133, 2002.
- [32] D. Lee and M. Spong. Passive bilateral teleoperation with constant time delay. *IEEE Transactions on Robotics*, 22(2):269–281, 2006.
- [33] H. Kawada and T. Namerikawa. Bilateral control of nonlinear teleoperation with time varying communication delays. In *Proceedings of the American Control Conference*, volume 189-194, 2008.
- [34] Y. Yasuyoshi, T. Imaida, and T. Yoshikawa. Bilateral control with energy balance monitoring under time-varying communication delay. *Proceedings of the IEEE International Conference on Robotics and Automation*, 3:2684–2689, 2000.
- [35] J.H. Ryu, D.S. Kwon, and B. Hannaford. Stable teleoperation with time-domain passivity control. *IEEE Transactions on Robotics and Automation*, 20(2):365–373, 2004.
- [36] Y. Ye, Y.J. Pan, and T. Hilliard. Bilateral teleoperation with time-varying delay: A communication channel passification approach. *IEEE/ASME Transactions on Mechatronics*, 18(4):1431–1434, 2013.
- [37] D.A. Lawrence. Stability and transparency in bilateral tele-operation. *IEEE Transactions on Robotics and Automation*, 9(5):624–637, 1993.
- [38] J. Lim, J. Ko, and J. Lee. Internet-based teleoperation of a mobile robot with force-reflection. In *Proceedings of 2003 IEEE Conference on Control Applications*, volume 1, pages 680–685, 2003.

- [39] Z. Xu, L. Ma, and K. Schilling. Passive bilateral teleoperation of a car-like mobile robot. In *Proceedings of 17th Mediterranean Conference on Control & Automation*, pages 790–796, 2009.
- [40] D. Lee, O. Martinez-Palafox, and M.W. Spong. Bilateral teleoperation of a wheeled mobile robot over delayed communication network. In *Proceedings of 2006 IEEE International Conference on Robotics and Automation*, pages 3298–3303, 2006.
- [41] Z. Zuo and D. Lee. Haptic tele-driving of a wheeled mobile robot over the internet: a pspn approach. In *49th IEEE Conference on Decision and Control*, pages 3614–3619, 2010.
- [42] Y.J. Pan, C. Canudas-de Wit, and O. Sename. A new predictive approach for bilateral teleoperation with applications to drive-by-wire systems. *IEEE Transactions on Robotics*, 22(6):1146–1162, 2006.
- [43] T. Ersal, R.B. Gillespie, M. Brudnak, J. L. Stein, and H. K. Fathy. Effect of Coupling Point Selection on Distortion in Internet- distributed Hardware-in-the-Loop Simulation. In *Proceedings of the American Control Conference*, pages 3096–3103, 2011.
- [44] T. Ersal, M. Brudnak, A. Salvi, J.L. Stein, Z. Filipi, and H.K. Fathy. Development and model-based transparency analysis of an Internet-distributed hardware-in-the-loop simulation platform. *Mechatronics*, 21(1):22–29, 2011.
- [45] T. Ersal, M. Brudnak, J.L. Stein, and H.K. Fathy. Statistical Transparency Analysis in Internet-Distributed Hardware-in-the-Loop Simulation. *IEEE/ASME Transactions on Mechatronics*, 17(2):228–238, 2012.
- [46] T. Ersal, M. Brudnak, A. Salvi, Y. Kim, J.B. Siegel, and J.L. Stein. An Iterative Learning Control Approach to Improving Fidelity in Internet-Distributed Hardware-in-the-Loop Simulation. *Journal of Dynamic Systems, Measurement, and Control*, 136(6):061012–061012–8, 2014.
- [47] X. Ge, M. Brudnak, J.L. Stein, and T. Ersal. A Norm Optimal Iterative Learning Control framework towards Internet-Distributed Hardware-In-The-Loop simulation. In *Proceedings of the American Control Conference*, pages 3802–3807, 2014.
- [48] D.P. Bertsekas, R.G. Gallager, and P. Humblet. *Data networks, vol. 2*. New Jersey: Prentice-Hall International, 1992.
- [49] Ford. Ford developing cross country automotive remote control, 2015. Accessed: 2018-4-8. URL <https://spectrum.ieee.org/cars-that-think/transportation/advanced-cars/ford-developing-cross-country-automotive-remote-control>.

- [50] J. Prokkola, P.H.J. Perälä, M. Hanski, and E. Piri. 3G/HSPA performance in live networks from the end user perspective. *IEEE International Conference on Communications*, 2009.
- [51] R. Liu, D. Kwak, S. Devarakonda, K. Bekris, and L. Iftode. Investigating Remote Driving over the LTE Network. In *Proceedings of the 9th International Conference on Automotive User Interfaces and Interactive Vehicular Applications*, pages 264–269, 2017.
- [52] Huawei. Huawei Demonstrates 5G-based Remote Driving with China Mobile and SAIC Motor, 2017. Accessed: 2018-4-8. URL <http://www.huawei.com/en/press-events/news/2017/6/5G-based-Remote-Driving>.
- [53] R. Oboe and P. Fiorini. A Design and Control Environment for Internet-Based Telerobotics. *The International Journal of Robotics Research*, 4(17):433–449, 1998.
- [54] J.C. Lane, C.R. Carignan, B.R. Sullivan, D.L. Akin, T. Hunt, and R. Cohen. Effects of Time Delay on Telerobotic Control of Neutral Buoyancy Vehicles. In *Proceedings of the 2002 IEEE International Conference on Robotics & Automation*, volume 3, pages 2874–2879, 2002.
- [55] National Robotics Engineering Center. Long Distance Teleoperation - Avatar, 2016. Accessed: 2018-4-8. URL <https://www.nrec.ri.cmu.edu/solutions/defense/other-projects/long-distance-teleoperation.html>.
- [56] W.S. Kim, B. Hannaford, and A.K. Fejczy. Force-reflection and shared compliant control in operating telemanipulators with time delay. *IEEE Transactions on Robotics and Automation*, 8(2):176–185, 1992.
- [57] J.P. Luck, P.L. McDermott, L. Allender, and D.C. Russell. An investigation of real world control of robotic assets under communication latency. In *Proceeding of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*, pages 202–209, 2006.
- [58] J. Storms and D. Tilbury. Equating user performance among communication latency distributions and simulation fidelities for a teleoperated mobile robot. In *IEEE International Conference on Robotics and Automation*, pages 4440–4445, 2015.
- [59] J. Storms. Modeling and Improving Teleoperation Performance of Semi-Autonomous Wheeled Robots by. *PhD Thesis, University of Michigan*, 2017.
- [60] P. Appelqvist, J. Knuuttila, and J. Ahtiainen. Development of an unmanned ground vehicle for task-oriented operation - Considerations on teleoperation and delay. In *IEEE/ASME International Conference on Advanced Intelligent Mechatronics, AIM*, pages 1–6, 2007.

- [61] O. Bodell and E. Gulliksson. Teleoperation of autonomous vehicle with 360° camera feedback. *Master Thesis, Chalmers University of Technology*, 2016.
- [62] J.E. Arnold and P.W. Braisted. *Design and evaluation of a predictor for remote control systems operating with signal transmission delays*, volume 2229. National Aeronautics and Space Administration, 1963.
- [63] S. Ammoun and F. Nashashibi. Real time trajectory prediction for collision risk estimation between vehicles. In *IEEE 5th International Conference on Intelligent Computer Communication and Processing*, pages 417–422, 2009.
- [64] P.F. Lima, M. Trincavelli, J. Mårtensson, and B. Wahlberg. Clothoid-Based Model Predictive Control for Autonomous Driving. In *European Control Conference (ECC)*, 2015.
- [65] R. Madhavan and C. Schlenoff. Moving object prediction for off-road autonomous navigation. In *Proceedings of the SPIE Aerosense Conference*, pages 134–145, 2003.
- [66] R. Rajamani. *Vehicle dynamics and control*. Springer Science & Business Media, 2011.
- [67] E. Fiala. Lateral forces on rolling pneumatic tires. *Zeitschrift VDI*, 96(29): 973–979, 1954.
- [68] H. Pacejka. *Tire and vehicle dynamics*. Elsevier, 2005.
- [69] S. Mathan, A. Hyndman, K. Fischer, J. Blatz, and D. Brams. Efficacy of a predictive display, steering device, and vehicle body representation in the operation of a lunar vehicle. In *Conference companion on Human factors in computing systems common ground - CHI '96*, pages 71–72, 1996.
- [70] Quantum Signal. Augmented Teleoperation and Shared Control, Accessed: 2018-4-8. URL http://www.quantumsignal.com/mobile_robotics/index.php.
- [71] A. Hosseini, F. Richthammer, and M. Lienkamp. Predictive haptic feedback for safe lateral control of teleoperated road vehicles in Urban Areas. In *IEEE Vehicular Technology Conference*, pages 1–7, 2016.
- [72] J. Baldwin, A. Basu, and H. Zhang. Panoramic video with predictive windows for telepresence applications. In *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C)*, volume 3, pages 1922–1927, 1999.
- [73] W.R. Mark, L. Mcmillan, and G. Bishop. Post-Rendering 3D Warping. In *Proceedings of 1997 Symposium on Interactive 3D Graphics*, pages 7–16, 1997.

- [74] D. Cobzas and M. Jagersand. Tracking and predictive display for a remote operated robot using uncalibrated video. In *Proceedings - IEEE International Conference on Robotics and Automation*, volume 1, pages 1847–1852, 2005.
- [75] Martin Jagersand, Adam Rachmielowski, David Lovi, and Neil Birkbeck. Predictive Display from Computer Vision Models. In *The 10th International Symposium on Artificial Intelligence, Robotics and Automation in Space I-SAIRAS*, pages 673–680, 2010.
- [76] C.C. Macadam. Understanding and Modeling the Human Driver. *Vehicle System Dynamics*, 40(1-3):101–134, 2003.
- [77] S. Vozar and D.M. Tilbury. Driver modeling for teleoperation with time delay. *IFAC-PapersOnline*, 19(3):3551–3556, 2014.
- [78] H. Mirinejad, P. Jayakumar, and T. Ersal. Modeling Human Steering Behavior during Path Following in Teleoperation of Unmanned Ground Vehicles. *Human Factors*, 2018.
- [79] N. Hogan. An organizing principle for a class of voluntary movements. *The Journal of Neuroscience*, 4(11):2745–2754, 1984.
- [80] T. Hiraoka, T. Kunimatsu, O. Nishihara, and H. Kumamoto. Modeling of driver following behavior based on minimum-jerk theory. In *12th World Congress on Intelligent Transport Systems*, pages 6–10, 2005.
- [81] Y. Amano, M. Hada, and S. Doi. A model of driver’s behavior in ordinary and emergent situations. *R&D Review of Toyota CRDL*, 33(1):23–30, 1998.
- [82] X. Ge, Y. Zheng, M.J. Brudnak, P. Jayakumar, J.L. Stein, and T. Ersal. Analysis of a Model-Free Predictor for Delay Compensation in Networked Systems. *Time Delay Systems*, pages 201–215, 2017.
- [83] Y. Zheng, M. J. Brudnak, P. Jayakumar, J. L. Stein, and T. Ersal. A Predictor Based Framework for Delay Compensation In Networked Closed-Loop Systems. *IEEE/ASME Trans. Mechatronics (In review)*, 2018.
- [84] Y. Zheng, M.J. Brudnak, P. Jayakumar, J.L. Stein, and T. Ersal. A Delay Compensation Framework for Predicting Heading in Teleoperated Ground Vehicles. *IEEE/ASME Trans. Mechatronics (Ready for submission)*, 2018.
- [85] Akshar Tandon, Mark J. Brudnak, Jeffrey L. Stein, and Tulga Ersal. An observer based framework to improve fidelity in internet-distributed hardware-in-the-loop simulations. In *ASME Dynamic Systems and Control Conference*, 2013.
- [86] Hale, J.K. *Theory of differential equations*. Springer Heidelberg, 1977.

- [87] T. Mori and E. Noldus. Stability criteria for linear differential difference systems. *International journal of systems science*, 15(1):87–94, 1984.
- [88] A.D. Myshkis. On solutions of linear homogeneous differential equations of the second order of periodic type with a retarded argument. *Matematicheskii Sbornik*, 70(1):15–54, 1951.
- [89] J.C. Lillo. Oscillatory solutions of the equation $y'(x) = m(x)y(x - n(x))$, 1969.
- [90] S. Wang, R. Nathuji, R. Bettati, and W. Zhao. Providing statistical delay guarantees in wireless networks. In *Proceedings of 24th International Conference on Distributed Computing Systems*, pages 48–55, 2004.
- [91] L.A. Montestruque and P. Antsaklis. Stability of model-based networked control systems with time-varying transmission times. *IEEE Transactions on Automatic Control*, 49(9):1562–1572, 2004.
- [92] J. Nilsson, B. Bernhardsson, and B. Wittenmark. Stochastic analysis and control of real-time systems with random time delays. *Automatica*, 34(1):57–64, 1998.
- [93] K. Gu, J. Chen, and V.L. Kharitonov. *Stability of time-delay systems*. Springer Science & Business Media, 2003.
- [94] E. Fridman. New lyapunov–krasovskii functionals for stability of linear retarded and neutral type systems. *Systems and Control Letters*, 43(4):309–319, 2001.
- [95] Y. Ariba and F. Gouaisbaut. An augmented model for robust stability analysis of time-varying delay systems. *International Journal of Control*, 82(9):1616–1626, 2009.
- [96] B.S. Razumikhin. Application of liapunovs method to problems in the stability of systems with a delay. *Automat, i Telemekh*, 21:740–749, 1960.
- [97] Hale, J.K. and Lunel, S.M.V. *Introduction to Functional Differential Equations*. Springer, New York, 1993.
- [98] W. Michiels, V. Van Assche, and S. Niculescu. Stabilization of time-delay systems with a controlled time-varying delay and applications. *IEEE Transactions on Automatic Control*, 50(4):493–504, 2005.
- [99] D. Huang and S.K. Nguang. State feedback control of uncertain networked control systems with random time delays. *IEEE Transactions on Automatic Control*, 53(3):829–834, 2008.
- [100] V.B. Kolmanovskii, T.L. Maizenberg, and J.P. Richard. Mean square stability of difference equations with a stochastic delay. *Nonlinear Analysis: Theory, Methods & Applications*, 52(3):795–804, 2003.

- [101] I. Kolmanovsky and T.L. Maizenberg. Mean-square stability of nonlinear systems with time-varying, random delay. *Stochastic analysis and Applications*, 19(2):279–293, 2001.
- [102] G.G. Yin and Q. Zhang. *Continuous-time Markov chains and applications: a two-time-scale approach*, volume 37. Springer Science & Business Media, 2012.
- [103] F. Wu, G.G. Yin, and L.Y. Wang. Stability of a pure random delay system with two-time-scale markovian switching. *Journal of Differential Equations*, 253(3): 878–905, 2012.
- [104] R.B. Israel, J.S. Rosenthal, and J.Z. Wei. Finding generators for markov chains via empirical transition matrices, with applications to credit ratings. *Mathematical finance*, 11(2):245–265, 2001.
- [105] S. Zahl. A markov process model for follow-up studies. *Human Biology*, 27(2): 90–120, 1955.
- [106] X. Ge, Y. Zheng, M.J. Brudnak, P. Jayakumar, J.L. Stein, and T. Ersal. Performance analysis of a model-free predictor for delay compensation in networked systems. *IFAC-PapersOnLine*, 48(12):434–439, 2015.
- [107] David L Mills. Adaptive Hybrid Clock Discipline Algorithm for the Network Time Protocol. *IEEE/ACM Transactions on Networking*, 6(5):505–514, 1998.
- [108] W.M. Griggs, B.D.O. Anderson, A. Lanzon, and M.C. Rotkowitz. Interconnections of nonlinear systems with “mixed” small gain and passivity properties and associated input–output stability results. *Systems and Control Letters*, 58(4):289–295, 2009.
- [109] P. Wu, M.J. McCourt, and P.J. Antsaklis. Experimentally determining passivity indices: Theory and simulation. *Interdisciplinary Studies in Intelligent Systems Lab Report*, page 002, 2013.
- [110] W.M. Griggs, B.D.O. Anderson, and A. Lanzon. A “mixed” small gain and passivity theorem in the frequency domain. *Systems and Control Letters*, 56(9-10):596–602, 2007.
- [111] Y. Zheng, M. Brudnak, P. Jayakumar, J. L. Stein, and T. Ersal. An Experimental Evaluation of a Model-Free Predictor Framework in Teleoperated Vehicles. *IFAC-PapersOnLine*, 49(10):157–164, 2016.
- [112] J. Liu, P. Jayakumar, J.L. Stein, and T. Ersal. A study on model fidelity for model predictive control based obstacle avoidance in high speed autonomous ground vehicles. *Vehicle System Dynamics*, 54(11):1629–1650, 2016.
- [113] C. Sierra, E. Tseng, A. Jain, and H. Peng. Cornering stiffness estimation based on vehicle lateral dynamics. *Vehicle System Dynamics*, 44:24–38, 2006.

- [114] MathWorks. MATLAB[®] Curve Fitting Toolbox[™] - Robust Least Squares, Accessed: 2018-4-8. URL <https://www.mathworks.com/help/curvefit/least-squares-fitting.html>.
- [115] Y. Zheng, M.J. Brudnak, P. Jayakumar, J.L. Stein, and T. Ersal. Human-in-the-loop Experimental evaluation of a Predictor Based Framework for Teleoperated Unmanned Ground Vehicles. *In preparation*, 2018.
- [116] MathWorks. MATLAB[®] Simulink[®] 3D Animation Toolbox[™], Accessed: 2018-4-8. URL <https://www.mathworks.com/products/3d-animation.html>.
- [117] Day Terry D. and L Daniel Metz. The simulation of driver inputs using a vehicle driver model. In *SAE Technical Paper*, number 2000-01-1313, 2000.
- [118] M Kamel Salaani, Gary J Heydinger, and Paul A Grygier. Closed Loop Steering System Model for the National Advanced Driving Simulator. *SAE Technical Paper*, pages 2004-01-1072, 2004.
- [119] A.B. Downey. Evidence for long-tailed distributions in the internet. In *Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement*, pages 229-241, 2001.
- [120] K. Salamatian and S. Vaton. Hidden Markov modeling for network communication channels. *ACM SIGMETRICS Performance Evaluation Review*, 29(1): 92-101, 2001.
- [121] M. Kalman and B. Girod. Modeling the delays of successively-transmitted internet packets. In *IEEE International Conference on Multimedia and Expo (ICME)*, volume 3, pages 2015-2018, 2004.
- [122] J.C. Bolot. Characterizing end-to-end packet delay and loss in the internet. *Journal of High Speed Networks*, 2(3):305-323, 1993.
- [123] M.E. Crovella and M.S. Taqqu. Estimating the Heavy Tail Index from Scaling Properties. *Methodology and Computing in Applied Probability*, 1:55-79, 1999.
- [124] G. Hooghiemstra and P. Van Mieghem. Delay distributions on fixed internet paths. *Delft University of Technology, report*, 20011020, 2001.
- [125] M. Karakas. *Determination of Network Delay Distribution over the Internet*. PhD thesis, The Middle East Technical University, 2003.
- [126] D.C. Wang, X. Fu, W. Ding, H. Li, N. Xi, and Y. Wang. Shifted gamma distribution and long-range prediction of round trip timedelay for Internet-based teleoperation. *IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 1261-1266, 2008.
- [127] S. Markose and A. Alentorn. The Generalized Extreme Value Distribution, Implied Tail Index, and Option Pricing. *The Journal of Derivatives*, 18(3): 35-60, 2011.

- [128] A.F. Jenkinson. The frequency distribution of the annual maximum (or minimum) values of meteorological elements. *Quarterly Journal of the Royal Meteorological Society*, 81(348):158–171, 1955.
- [129] P. Boehm, A.H. Ghasemi, S. O’Modhrain, P. Jayakumar, and R.B. Gillespie. Architectures for Shared Control of Vehicle Steering. *IFAC-PapersOnLine*, 49(19):639–644, 2016.
- [130] VectorNav. VectorNav VN-300 rugged Dual Antenna GNSS/INS, Accessed: 2018-4-8. URL <https://www.vectornav.com/products/vn-300>.
- [131] SainSmart. Wide angle FOV 160° 5-megapixel camera module for raspberry pi, Accessed: 2018-4-8. URL https://www.sainsmart.com/products/wide-angle-fov160-5-megapixel-camera-module-for-raspberry-pi?utm_medium=cpc&utm_source=googlepla&variant=46027760980&gclid=EAIaIQobChMI1bDu6pvD2gIVy8DCh1RsAt1EAQYASABEgJuNPD_BwE.
- [132] Navipedia. Transformations between ECEF and ENU coordinates, Accessed: 2018-4-8. URL http://www.navipedia.net/index.php/Transformations_between_ECEF_and_ENU_coordinates.
- [133] TOTMC. USB video capture adapter, Accessed: 2018-4-8. URL <http://www.totmc.com/product/USB-2-0-Video-Capture-Adapter-for-Windows.html>.
- [134] ISO. Double lane change scenario, Accessed: 2018-4-8. URL <https://www.iso.org/obp/ui/#iso:std:iso:3888:-1:ed-1:v1:en>.
- [135] A.H. Ghasemi, M. Johns, B.N. Garber, P. Boehm, P. Jayakumar, W. Ju, and R.B. Gillespie. Role negotiation in a haptic shared control framework. In *Adjunct Proceedings of the 8th International Conference on Automotive User Interfaces and Interactive Vehicular Applications*, pages 179–184, 2016.
- [136] H. Febbo. Nloptcontrol.jl. URL <https://github.com/JuliaMPC/NLOptControl.jl>.
- [137] A.H. Ghasemi, Y. Zheng, H. Febbo, A. Bhardwaj, J.L. Stein, T. Ersal, and R.B. Gillespie. Who’s the boss? arbitrating control authority between a human driver and automation system. *IEEE Transactions on Intelligent Transportation Systems (In preparation)*, 2018.
- [138] Y. Zheng, H. Febbo, A.H. Ghasemi, A. Bhardwaj, J.L. Stein, T. Ersal, and R.B. Gillespie. Compensating visual delays with model-free predictors in semi autonomous UGVs. *In preparation*, 2018.